



LIBRARY OF THE  
UNIVERSITY OF ILLINOIS  
AT URBANA-CHAMPAIGN

510.84

Il6r

no. 293-300

cop. 2



**CENTRAL CIRCULATION AND BOOKSTACKS**

The person borrowing this material is responsible for its renewal or return before the **Latest Date** stamped below. **You may be charged a minimum fee of \$75.00 for each non-returned or lost item.**

Theft, mutilation, or defacement of library materials can be causes for student disciplinary action. All materials owned by the University of Illinois Library are the property of the State of Illinois and are protected by Article 16B of Illinois Criminal Law and Procedure.

**TO RENEW, CALL (217) 333-8400.**

**University of Illinois Library at Urbana-Champaign**

JUN 28 1999

FEB 01 A.M.

When renewing by phone, write new due date below previous due date.

L162



Digitized by the Internet Archive  
in 2013

<http://archive.org/details/optimumnetworkde293baug>





062  
0.293  
op.2

OPTIMUM NETWORK DESIGN USING  
NOR AND NOR-AND GATES BY  
INTEGER PROGRAMMING

by

C. R. Baugh  
T. Ibaraki  
T. K. Liu  
S. Muroga

THE LIBRARY OF THE  
FEB 17 1969

January 10, 1969



DEPARTMENT OF COMPUTER SCIENCE  
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS





Optimum Network Design Using  
NOR and NOR-AND Gates by  
Integer Programming

C. R. Baugh  
T. Ibaraki  
T. K. Liu  
S. Muroga

January 10, 1969



510,84  
IL6r  
no 293-300  
eg 2 Abstract

In this paper the integer programming formulations<sup>[11] [12]</sup> of optimum network synthesis problems are demonstrated to be computationally feasible by actually obtaining all the optimum NOR gate networks and NOR-AND gate networks for three variable switching functions. (80 representative functions of equivalent classes by permutation and negation of variables.) The algorithm used for solving the integer programs is the implicit enumeration algorithm which was tailored to our problem with algorithmic and programming gimmicks.<sup>[9]</sup> The total computation time for these 80 functions on the IBM 360/75I was 110 minutes for NOR gate networks and 54 minutes for NOR-AND gate networks. The algorithm which was further modified by taking into consideration the inherent structures of NOR gate networks showed a significant improvement of computation time. With this second algorithm, all the optimum networks under fan-in and fan-out restrictions for the odd parity function which had not been computed before, were computed in 6 minutes and 30 seconds.

All the optimum NOR-AND gate networks for each of 80 three variable switching functions were also tabulated.

## 1. Introduction

Based on the integer linear programming approach discussed in the previous papers<sup>[11]</sup> <sup>[12]</sup>, optimum combinational networks of NOR gates and those of NOR-AND gates have been synthesized. This paper presents computational results. The integer programming algorithm used for solving those problems is the implicit enumeration method which is modified and it is described in detail elsewhere<sup>[9]</sup> <sup>[10]</sup>. The algorithm is further tailored to our problems by making use of the particular structures of our problems, improving the computational efficiency greatly over the first algorithm.

In this paper all the optimal networks of three variable switching functions with NOR gates and those with the combination of NOR-AND gates are exhausted by integer programming approach. The computation time on IBM 360/75I with the H level FORTRAN IV compiler for NOR networks for all three variable switching function (80 representative functions of equivalent classes by permutation and complementation of variables.) is 110 minutes, and it is further improved by the factor of about 10 times by using the above second algorithm. It takes 54 minutes for synthesizing optimum NOR-AND combination network for all three variable functions. This result may suggest the computational feasibility of integer programming approach and encourages further investigation.

## 2. Integer Programming and Implicit Enumeration

This section presents the outlines of the integer programming problem and implicit enumeration algorithm for solving it. For detailed description, see the references [6] [9] for example. The computer code used for the network synthesis is discussed in the reference [9]. It is a result of simpliciation and modification of the original implicit enumeration algorithm [1] [5] [6] [7] in order to improve computational efficiency.

An integer programming problem with  $n$  unknown variables and  $m$  constraints is in general stated as follows:

$$\begin{aligned} \text{Minimize} \quad & \bar{c} \bar{x} \\ \text{subject to} \quad & \bar{b} + A \bar{x} \geq 0, \end{aligned} \tag{2.1}$$

where  $\bar{c}$  is an  $n$ -dimensional vector of non-negative constants,  $\bar{b}$  is an  $m$ -dimensional vector of constants and  $A$  is an  $(m \times n)$  matrix of given coefficients, and  $\bar{x}$  is an  $n$ -dimensional vector of variables. In our case, all variables  $\bar{x}$  are integers which assume only 1 or 0. Sometimes this is referred to as the zero-one integer programming problem.

The implicit enumeration algorithm has been computationally proved to be one of the most efficient methods for solving this type of zero-one problem. It implicitly enumerates all the  $2^n$  solutions without explicitly and exhaustively examining all of them, and picks up the best feasible solution.

Let us start with several definitions. When all the variables in  $\bar{x}$  are assigned 1 or 0 it will be called a solution. If a solution satisfies the constraints  $A \bar{x} + \bar{b} \geq 0$ , it will be called a feasible solution and if not, an infeasible solution. A feasible solution that minimizes  $\bar{c} \bar{x}$  is an optimum feasible solution. A partial solution  $S$  is defined as

an assignment of binary values to a subset of the  $n$  variables. Any variables which are not assigned are called free variables. A completion of a partial solution  $S$  is a binary assignment to all free variables.

Let us outline the implicit enumeration algorithm as it is shown in figure 2.1. With a given partial solution  $S^0$  and the incumbent solution (the feasible solution having the smallest value of the objective function obtained thus far), the block entitled "CHK-IEQ" is entered. At this point, examine whether some of the free variables must be 1 or 0 if each inequality is to be satisfied. Scanning through the inequalities until no more free variables are assigned,  $S^0$  with these free variables assigned becomes a new partial solution  $S^1$ . Next the partial solution  $S^1$  is checked to determine which of the following 3 cases occurs.

(1) Feasible: The completion of  $S^1$  obtained by setting all free variables to 0 is found to be feasible. It is compared with the incumbent and the better of the two is maintained as the incumbent. The backtrack procedure is initiated to obtain a new partial solution  $S^2$  by changing some of the assigned variables according to a certain rule.\*

(2) Infeasible: If at least one inequality is not satisfied by  $S^1$  whatever binary values are assigned to the free variables, then  $S^1$  is discarded immediately by initiating the backtrack procedure. The backtrack procedure forms a new partial solution  $S^2$  from  $S^0$ .

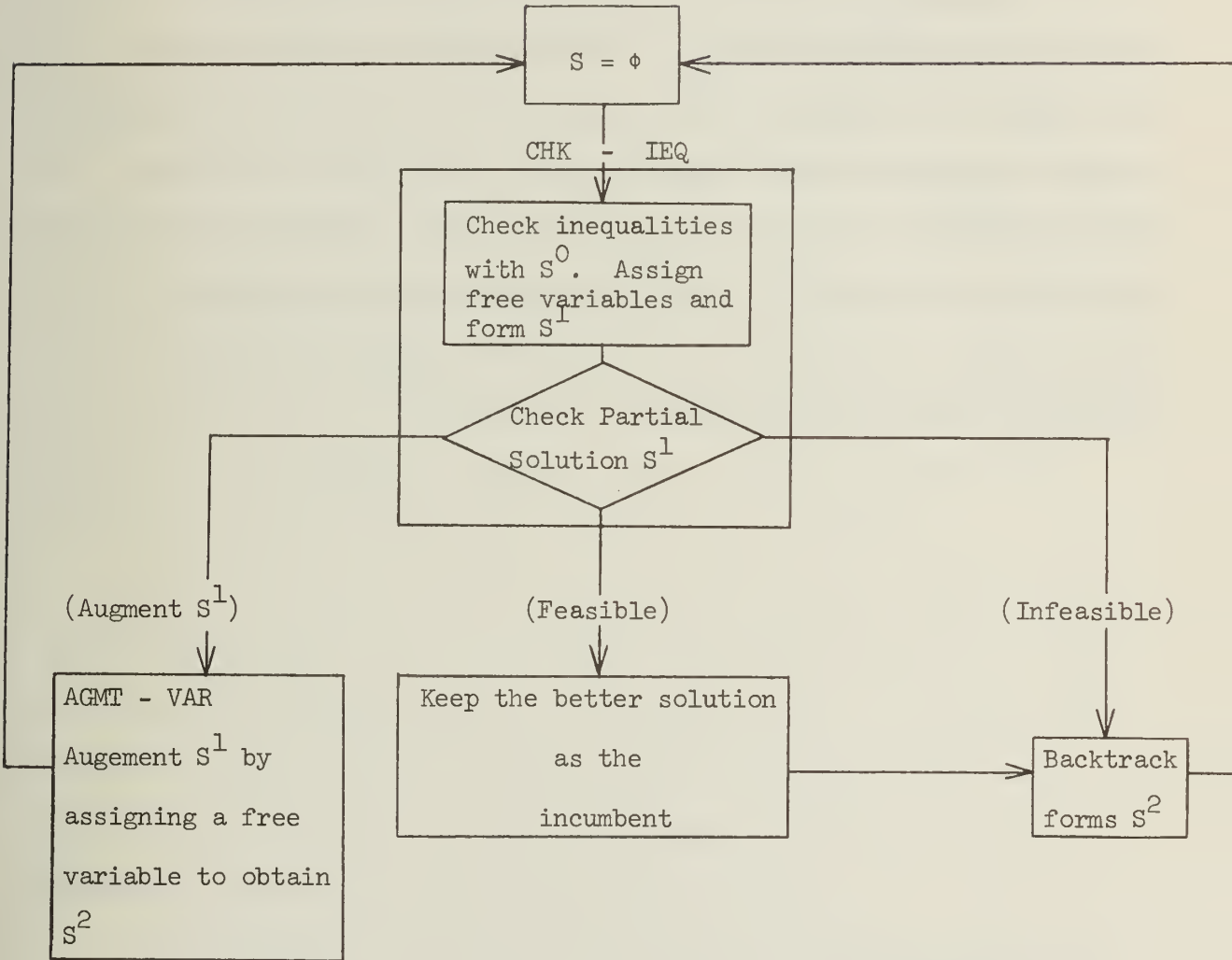
(3) Augment  $S^1$ : If neither of the above two cases occur, a free variable is assigned to 1, forming  $S^2$ . The choice of this variable greatly affects the convergence and it should be made according to the type of problem being solved.

---

\* The backtrack procedure was first proposed by Glover<sup>[7]</sup>. A detailed explanation can also be found in [6], [9] and will not be described in this report.

After replacing  $S^0$  with  $S^2$ , the entire procedure is repeated by reentering the block "CHK-IEQ".

Fig. 2.1 Implicit enumeration algorithm



By cycling through this procedure repeatedly, the computation results in the implicit enumeration of all possible solutions. When the computation terminates, the incumbent is the optimum solution. The checking procedure of each inequality such that one of cases (1), (2) and (3) is quickly identified is explained in [9]. The implicit enumeration algorithm converges in a finite number of steps, but the efficiency of the algorithm heavily depends on the nature of an individual problem. Our computational experience shows that tailoring the block labeled AGMT-VAR in Fig. 2.1 (the subroutine which augments the partial solution when (3) occurs)<sup>[9]</sup> to a given particular problem speeds up the convergence. Some aspects of our AGMT-VAR tailored to NOR gate network synthesis will be discussed in Section 4.



### 3. NOR Network Synthesis

All the optimum feed-forward NOR gate networks of three variable switching functions are realized by using the formulation described in [11]. Networks are optimum in the sense that the number of NOR gates in the networks is minimized first and the number of interconnections among gates and from external variables is minimized second. The same problem was already solved by L. Hellerman<sup>[8]</sup> by actually exhausting all the network configurations and then finding the best network among them for each function. When the problem is solved as an ordinary integer program, however, the computation time for all three variable switching functions (80 representative functions of equivalent classes by permutation and complementation of variables) is about 110 minutes\* on the IBM 360/751 with the H level FORTRAN IV compiler. This compares favorably with Hellerman's result consuming about 26 hours on the IBM 7090.

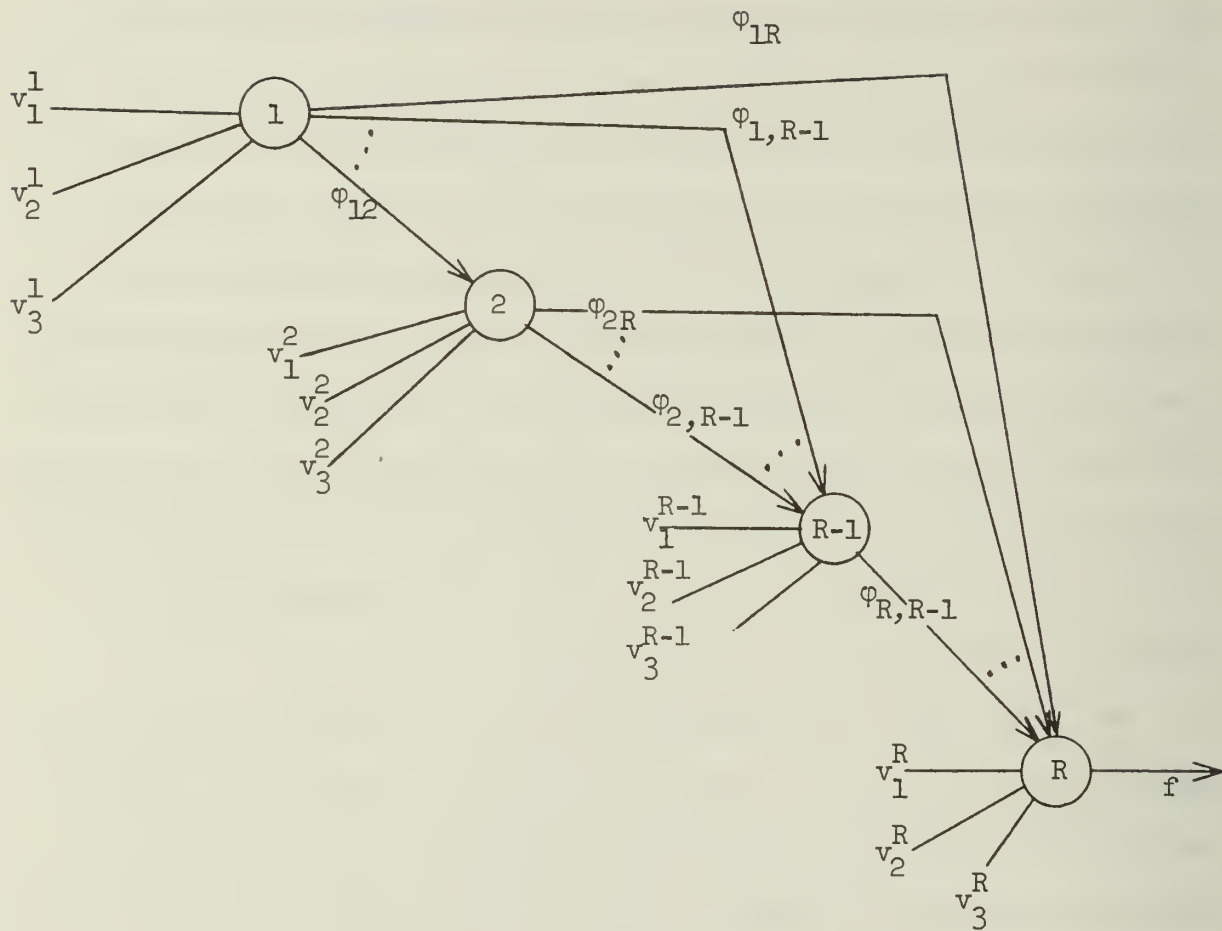
Now, the basic configuration for a NOR feed-forward network is shown in Figure 3.1. Let us explain the additional inequalities, which are introduced solely to reduce the computation time by precluding the redundant connections or by partially precluding networks which are equivalent by permutation of gates. The basic formulation of these additional inequalities is the same as that of [11].

The problems we solved have three external variables  $x_1, x_2, x_3$ . Let the connection from  $x_1$  to the  $i$ -th gate be  $v_1^i$ . Let the connection from the  $i$ -th gate to the  $k$ -th gate ( $i < k$ ) be  $\phi_{ik}$ . Since only completely specified functions are considered, each input vectors  $\vec{x} = (x_1, x_2, x_3)$  is numbered as  $\vec{x}^{(j)}$   $j = 1, 2, \dots, 8$  from  $\vec{x}^{(1)} = (0, 0, 0)$  through  $\vec{x}^{(8)} = (1, 1, 1)$ . The output value of the  $i$ -th gate for the  $j$ -th input vector  $\vec{x}^{(j)}$  is denoted as  $P_i^{(j)}$ . And let  $\rho_{ik}^{(j)}$  denote  $\phi_{ik} P_i^{(j)}$ .

---

\* In Section 4, this computation time is further reduced by designing a sophisticated AGMT-VAR algorithm.

Fig. 3.1 Feed-forward network



Inequalities for characterizing a feed-forward network with R NOR gates which realizes the function  $f(\vec{x})$  is as follows. [11]

For  $k = 1, 2, \dots, R-1$

$$\begin{aligned}
 - \sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} - \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq 0 - U (1 - P_k^{(j)}) \\
 \sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} + \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq 1 - U P_k^{(j)}
 \end{aligned} \tag{3.1}$$

$j = 1, 2, \dots, 8,$

where  $U$  is a sufficiently large positive number such that the upper inequality is non-restrictive if  $P_k^{(j)} = 0$  and the lower inequality is non-restrictive if  $P_k^{(j)} = 1$ .

For the last gate (the  $R$  th gate)

$$\begin{aligned}
 - \sum_{\ell=1}^3 v_{\ell}^R x_{\ell}^{(j)} - \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq 0 \text{ if } f(\vec{x}^{(j)}) = 1 \\
 \sum_{\ell=1}^3 v_{\ell}^R x_{\ell}^{(j)} + \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq 1 \text{ if } f(\vec{x}^{(j)}) = 0
 \end{aligned} \tag{3.2}$$

$j = 1, 2, \dots, 8.$

And in place of the relation  $\rho_{ik}^{(j)} = \varphi_{ik} P_i^{(j)}$ , we have the inequalities

$$- P_i^{(j)} - \varphi_{ik} + \rho_{ik}^{(j)} \geq -1 \tag{3.3}$$

$$P_i^{(j)} + \varphi_{ik} - 2 \rho_{ik}^{(j)} \geq 0$$

$k = 2, 3, \dots, R$

$i = 1, 2, \dots, k-1$

$j = 1, 2, \dots, 8.$

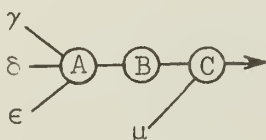
The procedure used in this paper to obtain all optimum solutions proceeds from  $R = 1$  and increases  $R$  by 1 each time until the feasible solution is reached. The objective function used is the number of interconnections

$$\sum_{k=1}^R \left( \sum_{i=1}^{k-1} \phi_{ik} + \sum_{\ell=1}^3 v_{\ell}^k \right), \quad (3.4)$$

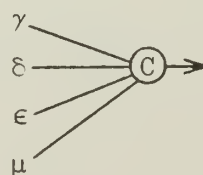
and therefore its minimization results in the minimum number of interconnections as the secondary objective. In other words, if we find the first feasible  $R$  and then exhaust all the solutions which minimizes the objective function (3.4), all the optimum solutions are obtained.

Seemingly the implicit enumeration algorithm has a tendency that the smaller the region of all solutions, the faster the convergence. Hence our effort was directed to preclude unnecessary solutions by adding extra inequalities so that the solution region becomes smaller without prohibiting any necessary optimum solutions. These additional inequalities essentially consist of two types; one is to preclude unnecessary network configurations and the other is to partially suppress the geometrically equivalent networks (i.e. those with permuted gates). They are listed in the following. Proofs are omitted.

(1) A NOR gate ( B in Fig. 3.2 (a)) which has only one input from another gate (A) with only one output, and which is not the output gate is not permitted in an optimum network, because the same



(a)



(b)

Fig. 3.2  
Cascaded  
connections

function can be realized with two fewer gates, as shown in Fig. 3.2 (b). Hence each gate except the last has at least one input from the external variables or at least two inputs from the other gates, which is expressed by an inequality:

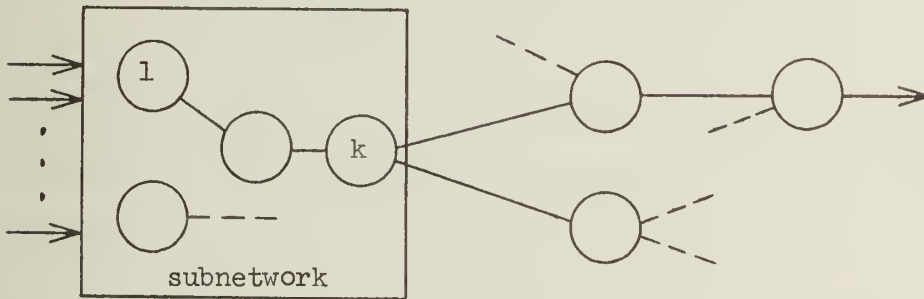
$$2 \sum_{\ell=1}^3 v_{\ell}^k + \sum_{i=1}^{k-1} \varphi_{ik} \geq 2 \quad (3.5)^*$$

$$k = 1, 2, \dots, R-1.$$

Note that this condition does not hold in general when the fan-in restriction is imposed.

(2) Consider any subnetwork which has only one gate which has outputs to gates not in the subnetwork (Fig. 3.3).

Fig. 3.3 Ordering of inputs



Assume the subnetwork consists of 1 through k. Let the k-th gate is the output gate of this subnetwork. Then we can order the interconnections to the k-th gate from the other gates in the subnetwork such that  $\varphi_{1k} \leq \varphi_{2k} \leq \dots \leq \varphi_{k-1,k}$ . In other words, inequalities:

$$\varphi_{1k} \leq \varphi_{2k} + \sum_{i=1}^{k-1} \sum_{j=k+1}^R \varphi_{ij}, \quad (3.6)$$

...

$$\varphi_{k-2,k} \leq \varphi_{k-1,k} + \sum_{i=1}^{k-1} \sum_{j=k+1}^R \varphi_{ij}$$

$$k = 3, 4, \dots, R,$$

\* The notation  $\sum_A^B$  is defined to be 0 when  $B < A$  holds.

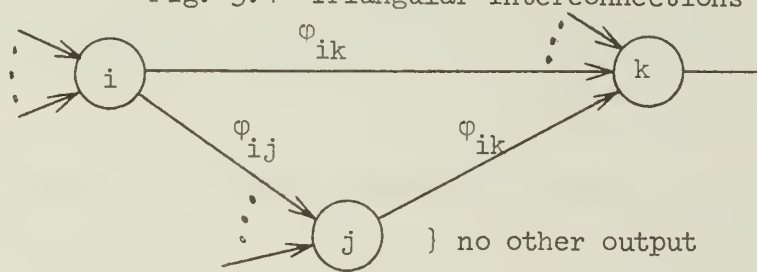
where the double sum  $\Sigma\Sigma$  is added in each inequality because it is assumed that the subnetwork has outputs only from one gate (in other words if all the gates in the subnetwork except the  $k$  th gate have no output to the outside of the subnetwork, i.e. if  $\Sigma\Sigma \varphi_{ij} = 0$ , then (3.6) yields

$$\varphi_{1k} \leq \varphi_{2k} \leq \dots \leq \varphi_{k-1,k}.$$

In the above discussion the subnetwork was assumed to consist of gate 1 through  $k$ . The extension to the general case in which the gates are not consecutively numbered starting from 1 is possible.

(3) Suppose that three gates are connected as shown in Fig. 3.4,

Fig. 3.4 Triangular interconnections



where the  $j$  th gate has no outputs except  $\varphi_{jk}$ . It is easily proved that if all of  $\varphi_{ij}$ ,  $\varphi_{ik}$ ,  $\varphi_{jk}$  are 1, the network is not optimum, thus introducing inequalities:

$$\varphi_{ij} + \varphi_{ik} + \varphi_{jk} \leq 2 + \sum_{\substack{t \neq k \\ t=j+1}}^R \varphi_{jt} \tag{3.7}$$

$$i < j < k \leq R.$$

Even if the  $i$  th gate in Fig. 3.4 is replaced by an external variable  $x_i$ , the above property is still true. Then from (3.7) we obtain:

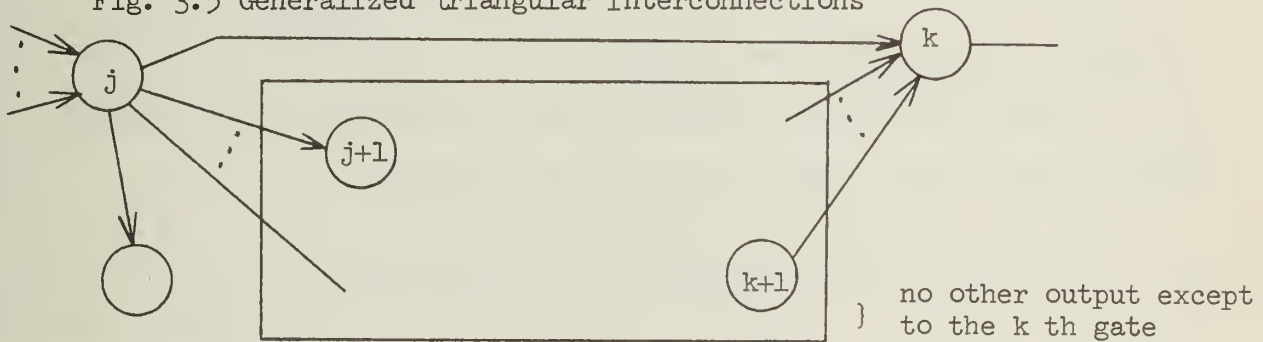
$$v_{\ell}^j + v_{\ell}^k + \varphi_{jk} \leq 2 + \sum_{\substack{t=j+1 \\ t \neq k}}^R \varphi_{jt} \quad (3.8)$$

$$j < k \leq R$$

$$\ell = 1, 2, 3.$$

(4) This condition is an extension of case (3). Consider any subnetwork which has no outputs except those to the  $k$ th gate and where the  $i$ th gate and the  $k$ -th gate is connected, as shown in Fig. 3.5. Assume that the subnetwork consists of the  $(i+1)$ st to  $(k-1)$ st gates.

Fig. 3.5 Generalized triangular interconnections



Then the interconnections from the  $i$ th gate to the subnetwork are all redundant and therefore can be deleted. This condition is written by an inequality:

$$\sum_{j=i+1}^{k-1} \varphi_{ij} \leq 0 + U \left( \sum_{h=1}^{k-i-1} \sum_{j=k+1}^R \varphi_{i+h,j} + (1-\varphi_{ik}) \right) \quad (3.9)$$

$$i = 1, 2, \dots, R-2$$

$$k = i + 2 \dots, R,$$

where  $U \geq \sum_{j=i+1}^{k-1} \varphi_{ij}$  for all  $i$  and  $k$ .

Even if the  $i$ -th gate is replaced by an external variable, the above property is still true. In this case the inequality is:



$$\sum_{j=i+1}^{k-1} v_{\ell}^j \leq 0 + U \left( \sum_{h=1}^{k-i-1} \sum_{j=k+1}^R \varphi_{i+h,j} + (1-v_{\ell}^k) \right) \quad (3.10)$$

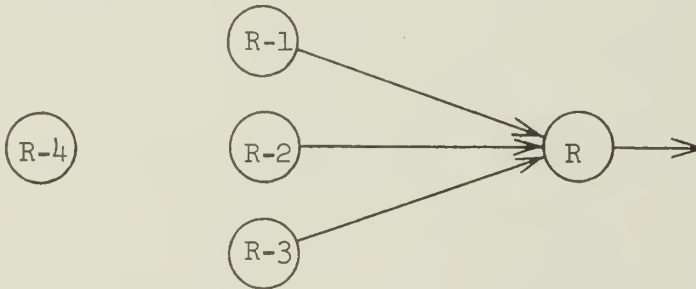
$$\begin{aligned} \ell &= 1, 2, 3 \\ i &= 1, 2, \dots, R-2 \\ k &= i+2, \dots, R, \end{aligned}$$

where  $U \geq \sum_{j=i+1}^{k-1} v_{\ell}^j$  for all  $i$  and  $k$ .

In the above discussion it was assumed that the subnetwork consists of the gates of the consecutive numbers (i.e. from the  $(i+1)$ st to  $(k-1)$ st gates) but an extension to the more general case where the subnetwork consists of gates of non-consecutive numbers is possible.

(5) A certain geometrical symmetry is also investigated. For example, Fig. 3.6 shows three gates connected to the last gate.

Fig. 3. 6 Symmetry of a subnetwork



In this configuration, it makes no difference to which of three gates the  $(R-1)$ ,  $(R-2)$ , and  $(R-3)$ , the  $(R-4)$ th gate is connected. Hence we can impose an ordering such that

$$\varphi_{R-4,R-1} \geq \varphi_{R-4,R-2} \geq \varphi_{R-4,R-3} \quad (3.11)$$

in order to preclude part of the gate configurations which are equivalent by permuting gates. This particular condition is represented by,



$$\varphi_{R-4,R-3} \leq \varphi_{R-4,R-2} + (1 - \varphi_{R-3,R}) + (1 - \varphi_{R-2,R}) + (1 - \varphi_{R-1,R}) \quad (3.12)$$

$$\varphi_{R-4,R-2} \leq \varphi_{R-4,R-1} + (1 - \varphi_{R-3,R}) + (1 - \varphi_{R-2,R}) + (1 - \varphi_{R-1,R})$$

Similar types of symmetry conditions are extensively considered and a number of such inequalities are employed in the actual computation.

However, the details of each type is not given here.

(6) If the interconnection between the  $i$ th and  $(i+1)$ st gates is known to be 0, these two gates are geometrically equivalent and their output connections can be ordered.

Hence we can first order their connections to the last gate as

$$\varphi_{i,R} \leq \varphi_{i+1,R} \quad (3.13)$$

If it turns out that  $\varphi_{i,R} = \varphi_{i+1,R}$  then the connections to the  $(R-1)$ th gate can be ordered as

$$\varphi_{i,R-1} \leq \varphi_{i+1,R-1}. \quad (3.14)$$

This argument continues until  $\varphi_{ik} \neq \varphi_{i+1,k}$  ( $k > i+1$ ) eventually holds. After that, no ordering on the connections is permitted. This sequential condition is expressed by the inequality:

$$\sum_{j=2}^{R-i} 2^{j-1} \varphi_{i,i+j} \leq \sum_{j=2}^{R-i} 2^{j-1} \varphi_{i+1,i+j} + U \varphi_{i,i+1}, \quad (3.15)$$

$$i = 1, 2, \dots, R-2.$$

(7) Each gate has at least one output, because the network is assumed to have exactly R gates. This condition is expressed by

$$\sum_{j=k+1}^R \varphi_{kj} \geq 1 \quad (3.16)$$

for every  $k = 1, 2, \dots, R-1$ .

The size of the problem is given in Table 3.1, both with a selected subset of the additional inequalities and without them.

TABLE 3.1 Size of NOR network formulation

R	Number of integer programming variables	Without additional inequalities		With additional inequalities
		Total number of inequalities	% of non-zero coefficients*	Total number of inequalities
2	23	40	14.58	-
3	52	88	7.12	-
4	90	152	4.31	169
5	137	232	2.91	265
6	193	328	2.11	415
7	258	440	1.60	716

\* For the function  $f(\bar{x}) = 0$ . For other functions, the sizes of figures are almost equal to those in the table.

The formulations of the problems may be characterized by the following properties:

(1) There are more inequalities than variables. Therefore the solution region is usually very small. In fact, many problems were found to be infeasible.

(2) The non-zero coefficients in the inequalities are fairly sparse. This feature was extensively utilized in our computer program of the implicit enumeration algorithm<sup>[9]</sup> in order to speed up the computation.

(3) The objective function has only coefficients of 0 or 1. This also simplifies the algorithm and is fully utilized<sup>[9]</sup>.

The implicit enumeration algorithm was used on the NOR synthesis problem and the results are given in TABLE 3.2. The algorithm was set to enumerate all optimal feasible solutions. It also assumed no fan-in and fan-out restrictions. The computation took approximately 110 minutes on the IBM 360/75I for all 80 representative functions. All networks were identical to Hellerman's.

We examined what are influential factors on the speed of our program. The effect of additional inequalities on speed-up was remarkable. Some problems tried for the R=5 formulation was speeded up by the factor of 5 in computation time.

As explained in[11] many types of network restrictions such as fan-ins and fan-outs can easily be added to the synthesis problem in the form of inequalities, without the necessity of changing or complicating the program. This is one of the advantages of the integer programming formulation. When the fan-in restriction is imposed, the restrictions

TABLE 3.2. Computational results of multiple optimum NOR network synthesis without fan-ins and fan-outs restrictions.

R	No. of function which require exactly R NORs.	Feasible				Infeasible	
		Average computation time per function (seconds)	Average number of iterations* per function	1st feasible solution (% of total iterations)	1st optimum solution	Average computation time per function (seconds)	Average No. of iterations* per function
1	3	.275	4	-----	-----	.075	2
2	5	.90	30	-----	-----	.49	27
3	8	3.59	104	23.19	35.92	1.99	73
4	17	42.26	954	11.15	58.48	30.52	843
5	23	982.00	15,908	23.81	64.40	-----	-----
6	15						
7	6						

Exhausted by hand

\* The number of iteration is defined as the number of times CHK-IEQ (see Fig. 2.1) is entered.

of (3.5) can no longer be included since it is possible to have a gate with a single input from another gate.\*

The execution times both with and without fan-in and fan-out restrictions, however, are on the average about the same for functions tested (although the computation time for each function is often different.).

Also by simply changing one statement in the program, we can modify the algorithm so that a single optimum solution is obtained rather than exhausting all the optimum solutions. For 6 gate functions tested, the computation time decreased by 10-20%.

Further considerable speed-up was gained by a more sophisticated modification of the implicit enumeration method by taking into account the physical structure of a network. This will be explained in the next section.

---

\* See also the argument on Fig. 4.2 in Section 4.

#### 4. Improvement of NOR Network Synthesis by the All-Interconnection Network Formulation

In the following computer program, we use a somewhat different integer programming formulation to characterize the feed-forward network, in addition to the reconstruction of AGMT-VAR. In other words, a network where an interconnection is permitted between every pair of gates (but the output gate has no output interconnections to other gate) is assumed and expressed in inequalities in the same way as [11]. Then inequalities are added to break loops which may result by solving the integer program with the above all-interconnection network. This modification is employed because, we can preclude a large number of partial solutions which are equivalent simply by permuting gates and which are difficult to preclude by other means.

This all-interconnection network formulation turns out to be very powerful for speeding up the NOR network synthesis.

The AGMT-VAR in Fig. 2.1 must be accordingly modified such that free variables of a partial solution be assigned, starting with the free variables associated with gate R and proceeding to gate R-1, then R-2, until reaching gate 1. This procedure also insures that there is no isolated subnetwork consisting of more than one gate. There may be various conceivable versions but we made the new AGMT-VAR based on Davidson's desirability order of gates, since it looked appropriate. This indicates the flexibility of our synthesis method by integer programming.

Recently Davidson<sup>[4]</sup> applied the "branch and bound method\*" to the NAND network synthesis, without describing the network by inequalities.

---

\* The implicit enumeration algorithm might be regarded as a synonym of the branch and bound method. But Davidson's algorithm and ours are quite different. Davidson's method is to construct a NAND network directly without inequalities and ours is to solve inequalities derived from the network.



He achieved a remarkable reduction of computation time, by judiciously making use of the intrinsic properties of a NAND gate. He defined the types of gates and determined empirically the desirability order of the types of gates. By checking all possible partial networks with very elaborate procedures, the optimality of the obtained network is guaranteed.

The types of gates and the desirability were incorporated with some modifications in our new AGMT-VAR. Our new computer program is more complex than the previous one. (The reduction of computation time was made at the expense of the complexity of the program.) It appears still simpler than Davidson's because the inequalities are still used and these inequalities provide simultaneously much information about the current partial solution without complicated program procedures (i.e. information about types of gates or covering condition which will be defined in the following).

The principle of this approach is based on the property of an NOR gate that :

$$\text{Property 1: If } P_k^{(j)} = 1, \text{ then}$$

$$\rho_{ik}^{(j)} = 0 \quad (4.1)$$

for all  $i = 1, 2, \dots, k-1$

must hold, and

$$\text{Property 2: } P_k^{(j)} = 0, \text{ then there exist at least one } i$$

$$(1 \leq i \leq k-1) \text{ or } \ell (1 \leq \ell \leq 3)$$

$$\text{such that } \rho_{ik}^{(j)} = 1 \text{ or } \bigvee_{\ell}^k x_{\ell} = 1$$

As defined before,  $P_k^{(j)}$  is the output value of the  $k$  th gate for the  $j$  th input vector,  $\rho_{ik}^{(j)}$  is the input value of the  $k$  th gate supplied from

the  $i$ -th gate for the  $j$ -th input vector, and  $v_{\ell}^k$  is the connection of the  $\ell$ -th external variable to the  $k$ -th gate.\*

Given a partial solution  $S$  in the course of the implicit enumeration, we have a partially constructed NOR network which may or may not lead to an optimum network eventually. Our modification of the algorithm is simply the reconstruction of the subroutine AGMT-VAR (see Fig. 2.1) such that it augments the partial solution with a variable selected according to the above property 2 of NOR gate in order to derive a reasonably good next partial solution from the current one. It is difficult to know what is good and there is no proof that the following procedure gives a good solution. However, our computational experience showed that it was considerably better than the AGMT-VAR which had been designed for general use and explained in the earlier sections.

Now a few definitions are given. A gate is said to be isolated if it has no output connected to other gate in the current partial solution. In other words, if all  $\phi_{ik}$ ,  $k \neq i$  are 0 or \*, the gate  $i$  is isolated. (The  $i$  th gate is currently isolated but could be connected by assigning 1 to a free variable.) If  $P_k^{(j)}$  is 0 in the current partial solution, let us associate with it one of the types which will be defined in the following. Let us define the types \*\*, "COV", "G\*" and so on as follows.

---

\* Note that during the computation the variables can take on 3 values 0 (zero), 1 (one), and \*(unassigned. i.e. a free variable)

\*\* These types are almost identical to Davidson's<sup>[4]</sup> but slightly different. And the number of types in this paper is fewer. G\* VC\*, GC\*, G\*C\*, NWG, approximately correspond to Davidson's EXP, VAR, FCN, EXF, NF, respectively. ISL and others which will be defined later are new concepts.



COV; If there exist  $i$  or  $\ell$  such that  $\rho_{ik}^{(j)} = 1$  or  $x_{\ell}^{(j)} v_{\ell}^k = 1$  (i.e. COVERed).

G\*; If there exists  $i$  such that  $\varphi_{ik} = 1$  and  $P_i^{(j)} = *$ .

(G\* stands for the Gate  $i$  with  $P_i^{(j)}$  unassigned.)

VC\*; If there exists  $\ell$  such that  $x_{\ell}^{(j)} = 1$  and  $v_{\ell}^k = *$

(VC\* stands for the Variable  $x_{\ell}^{(j)} = 1$  with Connection being \*.)

GC\*; If there exists  $i$  such that  $P_i^{(j)} = 1$  and  $\varphi_{ik} = *$ , when the

gate  $i$  is not isolated (Gate  $i$  with  $P_i^{(j)} = 1$  and Connection being \*)

G\*C\*; If there exists  $i$  such that  $P_i^{(j)} = *$  and  $\varphi_{ik} = *$ ,

where the gate  $i$  is not isolated (Gate  $i$  with  $P_i^{(j)} = *$  and

Connection being \*)

NWG; If there exists  $i$  such that  $P_i^{(j)} = *$  and  $\varphi_{ik} = *$ , where the gate

$i$  is isolated (NWG stands for NeW Gate.).

Since each  $P_k^{(j)}$  may satisfy more than one of the above conditions (i.e. each  $P_k^{(j)}$  may be associated with more than one type), let us order these types by desirability as

$$\text{COV, G*}, \text{VC*}, \text{GC*}, \text{G*C*}, \text{NWG.} \quad (4.2)$$

And the type of  $P_i^{(j)}$  is defined as the most desirable one among those

which satisfy the above definition, if  $P_k^{(j)} = 0$  in the current partial solution.

The motivation for defining these is to find a gate or external variables which leads to Property 2 of the earlier discussion. If Property 2 is satisfied or equivalently, if the type of  $P_k^{(j)}$  is COV,  $P_k^{(j)}$  is said covered. In the desirability order (4.2), G\*, for example, is more desirable than G\*C\* since it is more likely that the covering may be achieved by assigning 1 to \* of the gate which was already connected.

(i.e. no new gate added.) But it is rather difficult to see intuitively that  $G^*$  is more desirable than  $VC^*$ . (4.2) was determined empirically by Davidson [4] such that the computation time is minimized. In this sense the order of desirability may be susceptible to the type of gates with which the network is to be synthesized. In general, it can be determined only by trial-error programming experiments.

Note that by our implicit enumeration algorithm, every partial solution automatically satisfies Property 1 because otherwise it is rejected when the check of the partial solution is done in CHK-IEQ of Fig. 2.1.

As an auxiliary concept to the following definition of the type of partial network, let us define types of gates:

ISL: If the gate is isolated (ISoLated).

LTG: If the gate is not of ISL type and the outputs  $P_k^{(j)}$  for all  $j$ 's are 0 or \* (LaTently useful Gate in the sense that the covering condition may be met later.).

If a given gate (the  $k$  th) satisfies neither of the above definitions, then the least desirable type according to order (4.2) among the types of  $P_k^{(j)}$  for all  $j$ 's such that  $P_k^{(j)} = 0$  is defined as the type of gate  $k$ .

If the types of all the gates in the current partial solution are ISL, LTG or COV, then the current partial solution is feasible\*, i.e. the resulting feasible network is easily derived. If this is the case, the backtrack (Fig. 2.1) procedure is entered after comparing this feasible (network) solution with the incumbent.

---

\* In our design procedure where the number of gates in a network starts at 1 and then increases, the feasible case occurs only when the types of all the gates are COV.

Let the desirability of types of gates be defined as

$$\text{ISL, COV, LTG, G}^*, \text{VC}^*, \text{GC}^*, \text{C}^*\text{C}^*, \text{NWG.} \quad (4.3)$$

Then the type of partial network is defined as the least desirable<sup>\*</sup> type of gate among all gates.

Fig. 4.1 (a) shows a partial network corresponding to a current partial solution. Each  $P_k^{(j)} = 0$  is shown with its type for  $x^{(j)}$  of Fig. 4.1 (b). For example,  $P_3^{(1)}$  is of type  $\text{GC}^*$  because by setting  $\varphi_{23} = 1$ ,  $P_3^{(1)}$  can be covered.  $P_3^{(2)}$  is of type  $\text{G}^*\text{C}^*$  because of  $P_2^{(2)} = *$ . The type of each gate is shown. Obviously the type of this partial network is  $\text{G}^*\text{C}^*$  which is the type of gate 3.

Let us describe the procedure in the new AGMT-VAR. If the type of partial network is one of ISL, COV, LTG, the current partial solution is feasible because we use Procedure I in Section 5 of [11]. Otherwise let us identify  $P_k^{(j)}$  which defines the type of this partial network. In the definition of the types  $\text{G}^*$ ,  $\text{VC}^*$ ,  $\text{GC}^*$ ,  $\text{G}^*\text{C}^*$ , and NWG, free variables are associated with  $P_k^{(j)}$  such that if these free variables are set to 1,  $P_k^{(j)}$  is covered (e.g. in the case of  $\text{VC}^*$ ,  $P_k^{(j)}$  is covered if  $v_l^k$  is set to 1, i.e. if the corresponding free variable is set to 1.). This free variable is then specified so that  $P_k^{(j)}$  is covered. For example, Fig. 4.1 shows that  $P_3^{(2)}$  is identified according to the type of partial network.  $P_3^{(2)}$  is of type  $\text{G}^*\text{C}^*$  and can be covered by connecting gates 2 and 3, i.e. by setting  $\varphi_{23} = 1$  and specifying  $P_2^{(2)} = 1^{**}$ . Other types are similarly

\* The least desirable is chosen in the definition because unless the gate having the least desirable is made to satisfy the properties of NOR gates, the partial network is not feasible.

\*\* In actual programming, this is done by augmenting the current partial solution by  $\rho_{23}^{(2)} = 1$  rather than  $\varphi_{23} = 1$  and  $P_2^{(2)} = 1$ . Subsequently  $\varphi_{23}$  and  $P_2^{(2)}$  are set to 1 when the partial solution undergoes CHK-IEQ.

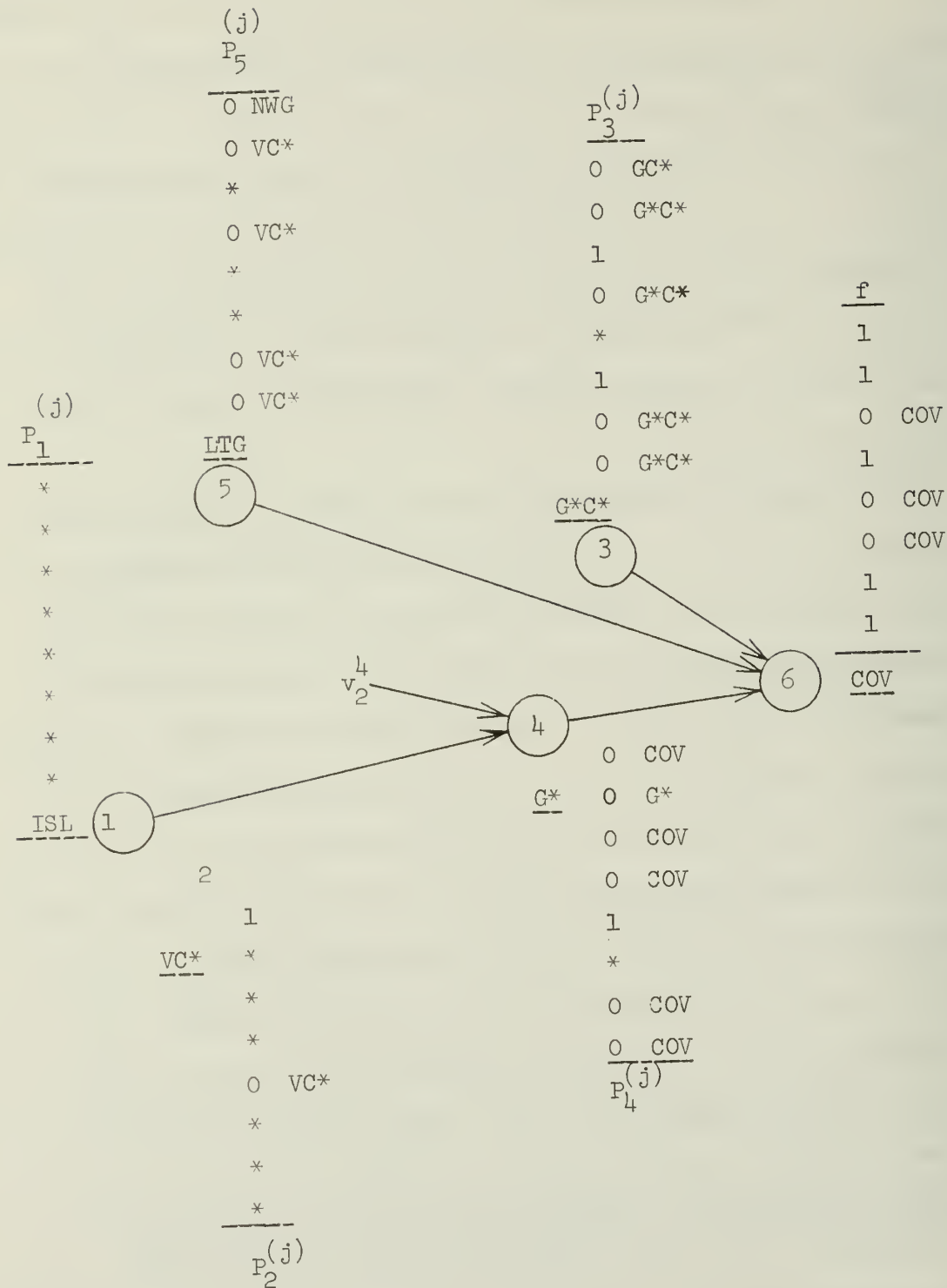


Fig. 4.1 (a) Example of types of  $P_k^{(j)} = 0$ , gates, and the partial network.

j	$x_1$	$x_2$	$x_3$
1	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

Fig. 4.1 (b) Assignment of values to  $\bar{x}^{(j)}$ .

treated: An appropriate  $\rho_{ik}^{(j)}$  in the case of  $G^*$ ,  $GC^*$ ,  $G^*C^*$  types or an appropriate  $v_{\ell}^k$  in case of  $VC^*$  is set to 1. However the type NWG is treated differently. If a gate is of type NWG, the isolated gate which has the largest gate number is identified. Let it be  $\gamma$ . Then  $\rho_{\gamma k}^{(j)}$  is set to 1 where  $P_k^{(j)}$  defines the type of partial network. Setting  $\rho_{\gamma k}^{(j)}$  to 1 will force  $\varphi_{\gamma k}$  and  $P_{\gamma}^{(j)}$  to 1, thus covering  $P_k^{(j)}$ . However if  $\rho_{\gamma k}^{(j)} = 0$  (i.e.  $\rho_{\gamma k}^{(j)}$  is not a free variable), all solutions with anyone of isolated gates connected to gate  $k$  have been investigated and do not need to be checked again. This follows from the fact that gate  $\gamma$  and any other isolated gate are equivalent with respect to the partial network (i.e. non-isolated part) since any isolated gate can be connected to any other gate. This approach eliminates the permutation of gate labeling which is inevitable with the feed-forward network. Of course, if we discover that  $\rho_{\gamma k}^{(j)}$  was already set to 0, then we can simply abandon the current partial solution and go to the backtrack procedure.

A comment is given on the case in which the type of the partial network is  $VC^*$ . If the types is  $VC^*$ , sometimes more than one external variable can be connected. In our algorithm, among all possible external variables, the external variable which covers the largest number of  $P_k^{(j)} = 0$  not yet covered is connected to the gate.

This new AGMT-VAR is used in place of AGMT-VAR shown in Fig. 2.1 and assigns a free variable to 1 according to the type of the partial network. Then CHK-IEQ is applied as before. Other part of algorithm are exactly the same as the general case discussed in the earlier sections and in [9], except that the following objective bound check is added to AGMT-VAR to further speed up the computation.

---

\* Note that  $\rho_{ik}^{(j)} = \varphi_{ik} P_k^{(j)}$ .



Given a partial solution, a lower bound of the objective function value is estimated according to the rules based on the following properties. If this bound exceeds the objective value of the incumbent, the current partial solution can not give any better solution than that and accordingly it is discarded. (The rest of computation for the current partial solution is skipped and backtrack is immediately applied). The bound estimation is programmed by using the following properties of the network:

- (1) Exactly R gates are assumed to be used in the network. In other words, each gate has at least one input connection and at least one output connection.
- (2) According to condition (1) of Section 3, each gate has either at least one input connection from external variable or at least two input connections from other gates.
- (3) The number of gates each of which is solely devoted to expressing  $\bar{x}_i$  (i.e. a gate has only one input which is an external variable  $x_i$ ) is at most three.
- (4) If the type of gate is GC\*, G\*C\* or NWG, the gate needs at least one more interconnection from another gate, as seen from the definitions of these types.
- (5) If the type of partial network is VC\*, the gate whose type defines the type of partial network needs at least one more connection from an external variable. If this single external variable does not cover all the  $P_k^{(j)}$  which are of type VC\*, we need to add at least one more external variable.

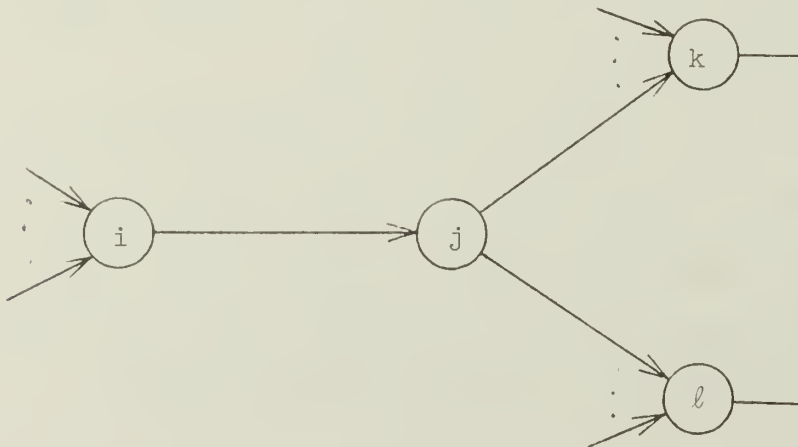
This bound estimation is included in AGMT-VAR and whenever it demonstrates that the current partial solution can not give any better feasible solution than the incumbent the backtrack procedure is performed immediately.

With all these modifications added, all 15 functions which can be realized with 6 gates were tested on the  $R=6$  formulation. The improvement was remarkable. The average number of iterations per function is 136.3 and the average computation time for each function is 4.99 seconds which is favorably compared with the result in Table 3.2 in Section 3 run with the general purpose AGMT-VAR, in which 954 iterations and 42.26 seconds on the average were necessary for each function.

If the fan-ins and fan-outs restrictions are considered, the rules based upon the above properties (2) and (3) must be modified. All other rules are unchanged. For example let us assume that each gate has maximum fan-ins and fan-outs of 3.

Let us examine Figure 4.2.

Fig. 4.2 Modification due to fan-in and fan-out restrictions





The single input to gate  $j$  is not allowed only if

$$\sum_{t=1}^3 (v_t^i + v_t^k) + \sum_{\substack{t=1 \\ t \neq i}}^{R-1} \varphi_{ti} + \sum_{\substack{t=1 \\ t \neq j \\ t \neq k}}^{R-1} \varphi_{tk} \leq 3 \quad (4.4)$$

$$\text{and } \sum_{t=1}^3 (v_t^i + v_t^l) + \sum_{\substack{t=1 \\ t \neq i}}^{R-1} \varphi_{ti} + \sum_{\substack{t=1 \\ t \neq j \\ t \neq l}}^{R-1} \varphi_{tk} \leq 3. \quad (4.5)$$

Property (3) also must be modified.

Using the rules based on these modified properties, the odd parity function  $x_1 \oplus x_2 \oplus x_3$  which requires 8 NOR gates when the fan-ins and fan-outs are limited to 3 was solved.

Since there is only one function to be solved, the structure of an optimum network for the function  $x_1 \oplus x_2 \oplus x_3$  is taken into consideration. In other words, this function is symmetric in all the three variables  $x_1, x_2$  and  $x_3$ , and accordingly the interconnections from  $x_l$  to the last gate are ordered as follows.

$$v_3^7 \geq v_2^7 \geq v_1^7 \quad (4.6)^*$$

and if  $v_{l+1}^7 = v_l^7$ , then

$$v_{l+1}^6 \geq v_l^6 \quad l = 1, 2. \quad (4.7)$$

---

\* It can be easily shown that the last gate (the 8th gate) has no external variables connected, if a given function can not be expressed as a conjunction of the product of literals and a disjunctive expression.

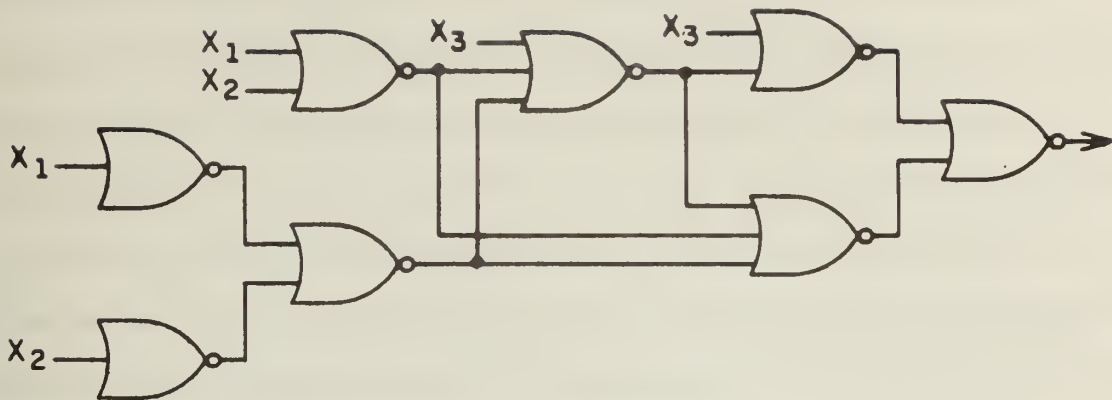
((4.7) could be continued to the gate numbers lower than 6. But the continuation was not incorporated in our program.) Also two interconnections  $\varphi_{68}$  and  $\varphi_{78}$  are assumed to be 1. This is guaranteed by the fact that the networks which are constructed by adding one gate to the output of optimum 7 NOR gate networks of  $\overline{x_1 \oplus x_2 \oplus x_3}$ , give no better realization of  $x_1 \oplus x_2 \oplus x_3$  than those obtained by solving the 8 gate formulation.

The problem has 395 variables and 1012 inequalities including additional inequalities. All optimum solutions are shown in Fig. 4.3. The network (c) of Fig. 4.3 is the same as that found by Tangiguchi et al. [13]

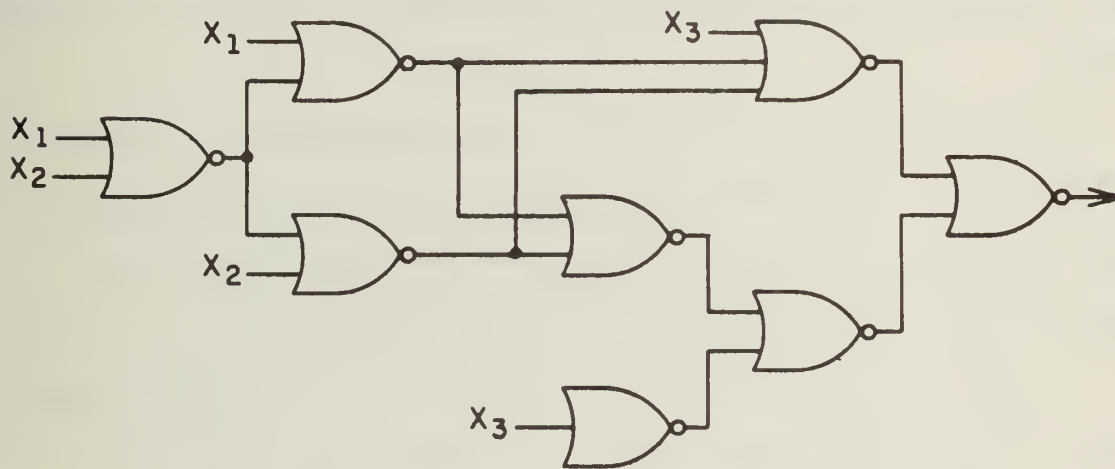
The computation took 4822 iterations with network (a) occurring at the 1368th iteration, network (b) at 1647th iteration, and solution (c) at the 3052th iteration. The total computation took less than 6 minutes and 30 seconds on the IBM 360/75I. When the program was modified so that only one of the optimum solution is to be found, the time was reduced to 5 minutes 15 seconds. These computation times are a considerable reduction with respect to the results of Table 3.2 in Section 3.

The reduction of the computation is due to the desirability order of types and the all-interconnection formulation. The all-interconnection network formulation appears to contribute more to the reduction than the desirability order. The desirability order probably has to be changed when different types of gates are to be used.

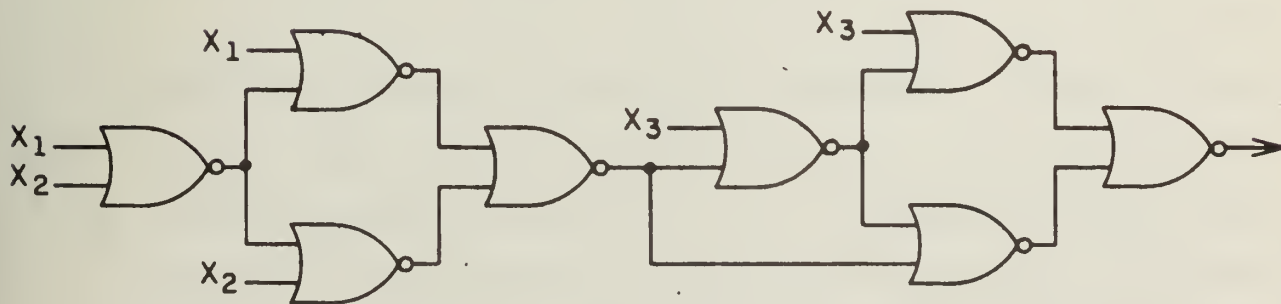
Our computation times are about the same as Davidson's, though exact comparison is very difficult because computers used are different and simple programming gimmicks may further make changes in computation times.



(a)



(b)



(c)

Figure 4.3: All optimum networks for  $x_1 \oplus x_2 \oplus x_3$

Probably one of the advantages of our approach is ease of programming effort. Since we can fully use the information associated with variables of the inequalities, the procedure to determine the types of  $P_k^{(j)}$ , gates and eventually the partial network is fairly straightforward and simple. Also Property 1 of the NOR gate is automatically taken care of by the CHK-IEQ routine, thus eliminating the procedure for checking this property. Another advantage is the fixed amount of storage needed throughout the computation. In Davidson's "branch and bound algorithm" the amount of storage often grows as the computation proceeds. Thus additional programming must be done in order to recalculate needed information or to use secondary storage.

Because of great improvement of the all-interconnection network formulation over the approach of Section 3, improvement in computation time even in the case of multiple-output network design also can be expected with the all-interconnection network formulation.

The accurate comparison of the exhaustive method, Davidson's and ours, in terms of the program complexity, computational efficiency, ease of programming and all other aspects, is difficult\* at this stage because a number of factors should be considered for the comparison and the data obtained so far is not sufficient. Here we tried this new approach to show the flexibility of the implicit enumeration algorithm. In other problems also, it is quite likely that we can achieve a great improvement by considering the intrinsic properties of the problem and incorporating appropriate modifications of the algorithm. Some examples of this approach will be found elsewhere [9].

---

\* Our integer programming approach can be extended to the logical design of an optimum sequential network and to the diagnosis of a network, different from Davidson's.

## 5. NOR-AND Network Synthesis

One of the important advantages of integer programming formulation is that we can solve a wide variety of problems by simply changing the set of inequalities. Networks composed of NOR and AND gates are represented in inequalities according to the formulation in [12]. Of course the optimum networks are also interesting in their own right. All the optimal networks for 80 three variable switching functions are tabulated under the same optimality criterion as NOR networks.

All the notations in this formulation are the same as NOR network case except that we have to introduce new variables indicating whether each gate is NOR or AND. These variables are

$$\theta_k, k=1, 2, \dots, R. \quad (5.1)$$

$\theta_k = 1$  (0) shows that the  $k$ th gate is NOR (AND).

The basic set of inequalities describing a feed-forward network of  $R$  gates is as follows. This formulation is found in [12].

For  $k = 1, 2, \dots, R-1$

$$\begin{aligned} \sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} + \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq \sum_{\ell=1}^3 v_{\ell}^k + \sum_{i=1}^{k-1} \phi_{ik} - U(1-P_k^{(j)}) - U\theta_k \\ -\sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} - \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq -\sum_{\ell=1}^3 v_{\ell}^k - \sum_{i=1}^{k-1} \phi_{ik} + 1 - U P_k^{(j)} - U\theta_k \\ -\sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} - \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq 0 - U(1-P_k^{(j)}) - U(1-\theta_k) \\ \sum_{\ell=1}^3 v_{\ell}^k x_{\ell}^{(j)} + \sum_{i=1}^{k-1} \rho_{ik}^{(j)} &\geq 1 - U P_k^{(j)} - U(1-\theta_k) \end{aligned} \quad (5.2)$$

$$j = 1, 2, \dots, 8.$$

It may be easily seen that the first two inequalities are for an AND gate and the others for a NOR gate, depending on the value of  $\theta_k$  associated with these inequalities.

For the last gate, if  $f(\bar{x}^{(j)}) = 1$ ,

$$\begin{aligned} \sum_{l=1}^3 v_l^R x_l^{(j)} + \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq \sum_{l=1}^3 v_l^R + \sum_{i=1}^{R-1} \rho_{iR} - U \theta_R \\ -\sum_{l=1}^3 v_l^R x_l^{(j)} - \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq 0 - U (1-\theta_R) \end{aligned} \quad (5.3)$$

and if  $f(\bar{x}^{(j)}) = 0$ ,

$$\begin{aligned} -\sum_{l=1}^3 v_l^R x_l^{(j)} - \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq -\sum_{l=1}^3 v_l^R - \sum_{i=1}^{R-1} \rho_{iR} + 1 - U \theta_R \\ \sum_{l=1}^3 v_l^{(j)} x_l^{(j)} + \sum_{i=1}^{R-1} \rho_{iR}^{(j)} &\geq 1 - U(1-\theta_R) \end{aligned} \quad (5.4)$$

for  $j = 1, 2, \dots, 8$ .

The relation  $\rho_{ik}^{(j)} = \varphi_{ik} P_i^{(j)}$  is expressed by

$$\begin{aligned} -P_i^{(j)} - \varphi_{ik} + \rho_{ik}^{(j)} &\geq 1 \\ P_i^{(j)} + \varphi_{ik} - 2\rho_{ik}^{(j)} &\geq 0 \end{aligned} \quad (5.5)$$

$$k = 2, 3, \dots, R$$

$$i = 1, 2, \dots, k-1$$

$$j = 1, 2, \dots, 8.$$

Additional inequalities are also incorporated to reduce the computation time. All of those additional inequalities which were used in the all NOR network case are included except the geometrical symmetry restrictions given by (5) of Section 3. In addition the following two types of inequalities were added.



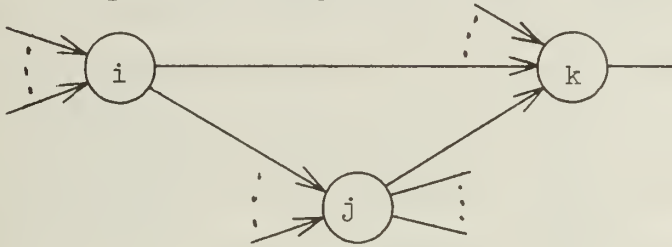
(1) Input interconnections

Each AND gate has at least two input interconnections. This constraint is given by

$$\sum_{\ell=1}^3 v_{\ell}^k + \sum_{i=1}^{k-1} \rho_{ik} \geq 2 - \theta_k \quad (5.6)$$

$k = 1, 2, \dots, R.$

Fig. 5.1 Triangular interconnection



(2) Triangular interconnections

Again consider three gates connected together as shown Fig. 5.1. Contrary to the NOR network case, if either of the gates j and k is AND gate, at least one of the three connections  $\varphi_{ij}$ ,  $\varphi_{ik}$ ,  $\varphi_{jk}$  must be 0 even if the j-th gate has outputs other than  $\varphi_{jk}$ . This condition is given by

$$\varphi_{ij} + \varphi_{jk} + \varphi_{ik} \leq 2 + \theta_j \quad (5.7)$$

$$\varphi_{ij} + \varphi_{jk} + \varphi_{ik} \leq 2 + \theta_k$$

$$i < j < k \leq R.$$

When the i th gate is replaced by an external variable,

$$v_{\ell}^j + v_{\ell}^k + \varphi_{jk} \leq 2 + \theta_j \quad (5.8)$$

$$v_{\ell}^j + v_{\ell}^k + \varphi_{jk} \leq 2 + \theta_k$$

$$j < k \leq R$$

$$\ell = 1, 2, 3.$$

Together with these additional inequalities (some of additional inequalities based on the properties which have been discussed are not incorporated because of their excessive number.), all the optimal NOR-AND combination networks for each function are solved. The entire computation took about 54 minutes on the IBM 360/75I, using a program which includes the general purpose AGMT-VAR. All functions are realized with not more than six gates. The size of each problem and computation time are listed in Tables 5.1 and 5.2. The computation time is graphically given in Fig. 5.2 as well as the NOR network case.

The effect of additional inequalities was remarkable. For the  $R=3$  case the computation time was reduced 6 times by incorporating the additional inequalities; and for the  $R=4$  case 18 times. Incorporation of other additional constraints such as the constraint that each AND gate should have at least one NOR gate connected to its output, will reduce the computation time further, though these were not actually tried.

Also contrary to NOR gate network, no effort was made to improve the computational efficiency of the integer programming algorithm. Of course a significant reduction of computation time can be expected by modifying the algorithm as discussed in Section 4.

A comparison of computation time with NOR gate networks is shown in TABLE 5.3 where the ratio of the computation time of the NOR-AND network case to that of NOR network case discussed in Section 3 is listed.



R	No. of functions realizable with R gates (representative function of equivalent classes)	Size of integer programming problems			% of non-zero entries in basic inequalities*
		No. of variables	No. of basic inequalities	No. of total inequalities including additional inequalities	
3	15	55	128	135	8.37
4	29	94	208	287	5.22
5	14	142	304	450	3.60
6	5	199	416	657	2.65

Table 5.1 Statistics of optimum NOR-AND network formulation

\* For the function which is identically 1.

TABLE 5.2 Computation time and number of iterations of NOR-AND network synthesis for all 80 three variable functions

R	No. of feasible functions	Feasible		Infeasible	
		Average computation time per function (seconds)	Average No. of iterations per function	Average computation time per function (seconds)	Average No. of iterations per function
0	3				
1	5				
2	9				
3	15	0.82	40	0.54	36
4	29	3.62	115	3.11	110
5	14	34.9	844	28.7	745
6	5	475.0	8734	-----	-----

Exhausted by hand

TABLE 5.3 Ratio of computation time

R	Computation Time		Iterations	
	Feasible	Infeasible	Feasible	Infeasible
4	4.0	6.3	3.8	4.1
5	9.8	14.4	8.1	10.2
6	11.3	----	9.2	----

On the average a function can be realized by NOR-AND combination network with 0.85 fewer gates than that of the NOR network case. The number of interconnections is also reduced by 1.5 on the average. All the optimal networks for three variable function are tabulated in Appendix A.

The tabulation also gives all the optimal networks by NAND-OR combination by the following procedure: (1) take the dual of a given function  $f$ , (2) find optimal networks for the dual function  $f^d$  by NOR-AND combination and (3) replace NOR by NAND, and AND by OR. These are optimal realizations of  $f$  by NAND-OR combination.

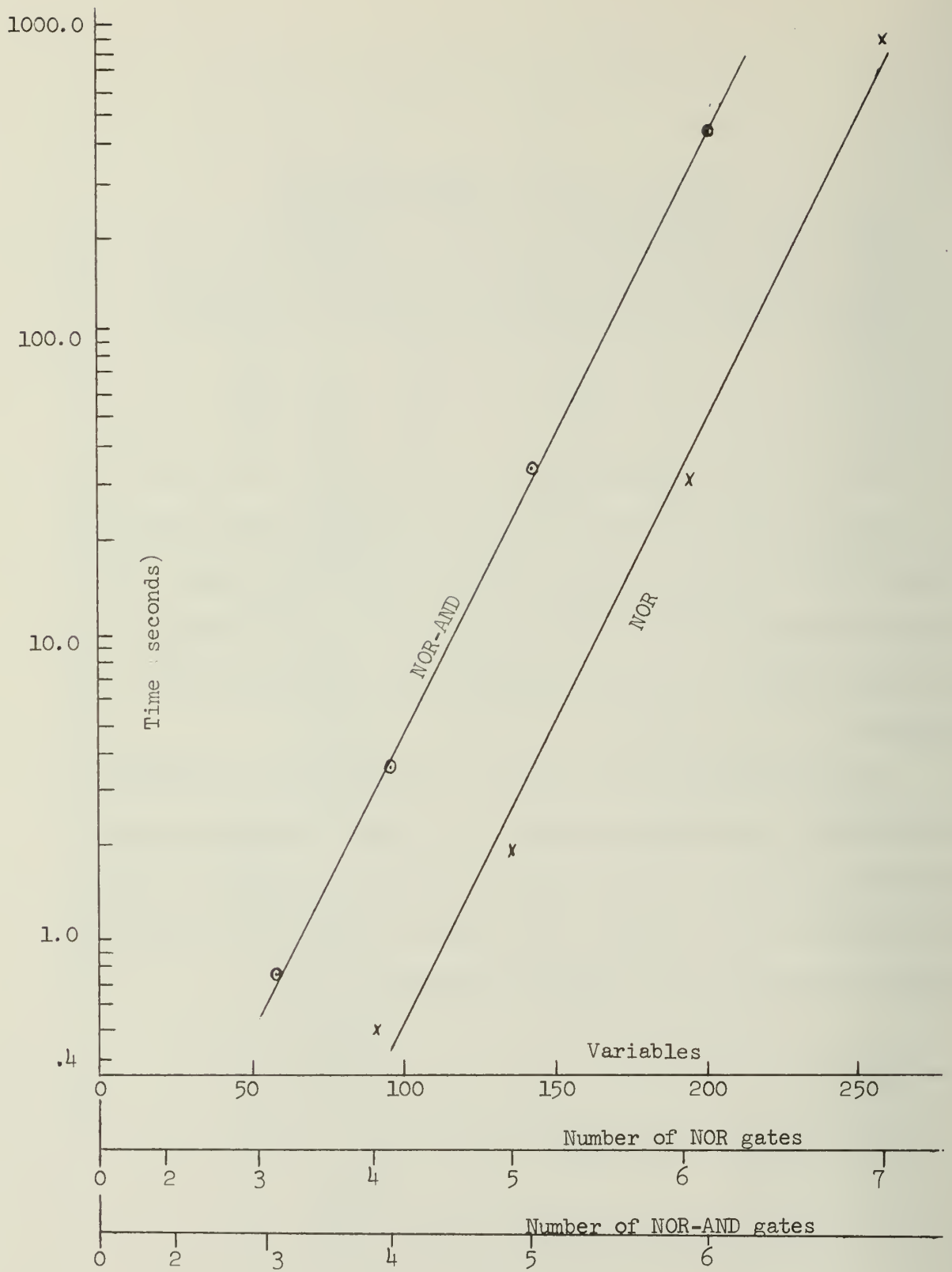


Fig. 5.2 Average computation time per function for NOR and NOR-AND synthesis

## 6. Conclusion

Two logical design problems, one with NOR gates only and the other with NOR-AND combination, formulated as integer linear programs were solved by using the implicit enumeration algorithm. NOR gate network synthesis is very important from an engineering view point and has attracted much attention. Integer programming approach to this problem is proved to be computationally feasible and is faster than Hellerman's exhaustive method. Advantages of integer programming approach include its versatility and simplicity to handle a variety of gate types, a variety of network restrictions such as maximum fan-ins restriction, and also various objectives, without changing the algorithm.

Simply changing part of the algorithm i.e. the AGMT-VAR, we can have a program of high speed at the sacrifice of the ease of programming, a program of simplicity and generality at the sacrifice of speed, and a wide range of programs between these two extremes.

Due to the principle of the implicit enumeration algorithm, we can modify and augment the algorithm so that the intrinsic properties of the problem can be fully used. Section 4 explored this possibility along with Davidson's work and obtained a improved result in NOR gate network synthesis. Furthermore three networks for the odd parity function were proven to be optimum.

By simply changing the inequalities, different problems can be solved. NOR-AND gate networks were thus synthesized by using different set of inequalities. Also we believe that this is the first tabulation of all the optimum networks with NOR and AND gates for all three variable functions.

## References

- [1] E. Balas, "An additive algorithm for solving linear programs with zero-one variables," Operations Research, vol. 13, no. 4 pp. 517-549, July-August 1965.
- [2] M. A. Breuer, "Implementation of threshold nets by integer linear programming," IEEEEC, vol. EC-14, no. 6, pp. 950-952, December 1965.
- [3] S. H. Cameron, "The generation of minimal threshold nets by an integer program," IEEEEC, vol. EC-13, no. 3, pp 299-302, June 1964.
- [4] E. S. Davidson, "An algorithm for NAND decomposition of combinational switching function," Ph.D. dissertation, Department of Electrical Engineering and Coordinated Science Laboratory, University of Illinois, 1968.
- [5] B. Fleischman, "Computational experience with the algorithm of Balas," Operations Research, vol. 15, no. 1, pp. 153-155, January - February 1967.
- [6] A. M. Geoffrion, "Integer programming by implicit enumeration and balas' method," SIAM Review, vol. 9 no. 2, pp 178-190, April 1967.
- [7] F. Glover, "A multiphase-dual algorithm for the zero-one integer programming problem," Operations Research vol. 13, no. 6, pp 879-919, November-December 1965.
- [8] L. Hellerman, "A catalog of three - variable OR-invert and AND-invert logical circuits," IEEEEC, vol. EC-12 no. 3 pp. 198-223, June 1963
- [9] T. Ibaraki, T. K. Liu, C. R. Raugh, and S. Muroga, "Implicit enumeration program for zero-one integer programming," to be published.
- [10] T. K. Liu, "A code for zero-one integer linear programming by implicit enumeration," Master thesis, Department of Computer Science, University of Illinois, 1968.
- [11] S. Muroga and T. Ibaraki, "Logical design of an optimum network by integer linear programming - Part I," Report No. 264, Department of Computer Science, University of Illinois, July 1968.
- [12] S. Muroga and T. Ibaraki, "Logical design of an optimum network by Integer Linear programming - Part II," Report No. 289, Department of Computer Science, University of Illinois, December 1968.
- [13] K. Taniguchi, N. Tokura, T. Kasami and H. Ozaki, " Logical functions realizable by a planar NAND network," The Trans. of Electronics and Communication Engineers of Japan , vol. 51-C, no. 2, pp. 59-65, February 1968.

Appendix A: Tabulation Of Optimum NOR-AND Networks

Here listed are all the optimum networks for each of all functions of up through three variables, using NOR and/or AND gates.

The networks which have the minimum number of interconnections and connections among the networks with the minimum number of gates are chosen as optimal network.

The procedure for obtaining the optimum network diagram for a given function follows that of Hellerman's [8]. A function can be represented by a truth table as shown below, by specifying the values of  $f_1, f_2, \dots, f_8$  where  $f_1, \dots, f_8$  show the values of the given  $f$  for input vectors in the same rows,  $a, b$  and  $c$  denote the variables of  $f$ .

	a	b	c
$f_1$	0	0	0
$f_2$	0	0	1
$f_3$	0	1	0
$f_4$	0	1	1
$f_5$	1	0	0
$f_6$	1	0	1
$f_7$	1	1	0
$f_8$	1	1	1

Let us write eight binary numbers  $f_1, \dots, f_8$  as follows.

$$\underbrace{f_8 \ f_7}_{0_2} \quad \underbrace{f_6 \ f_5 \ f_4}_{0_1} \quad \underbrace{f_3 \ f_2 \ f_1}_{0_0}$$



grouping the  $f_i$ 's as shown, we obtain three octal number  $O_2 O_1 O_0$ . This octal number is used throughout Appendix to identify a function.

In Tables A2 and A3, only representatives of equivalence class obtained by permutation of variables are listed, reducing 256 functions into 80 representatives. The representative of a given function  $f$  can be easily derived by using Table A1, which is taken from Hellerman [8]. The procedure is illustrated by an example.

Suppose that the function  $f$  has the number 321. The entry for this number in Table A1 is  $5*213$ . This means that  $f$  is equivalent to function 213 by applying permutation 5 which is (ac) as also shown in Table A1. Table A2 shows that function 213 has optimum networks with network numbers 49 and 56. Then Table A3 gives us the actual optimum networks of function 213. To obtain optimum networks of 321, we apply the inverse of permutation 5, which is also (ac), to these networks in Table A3.

Twelve of the 80 three-variable functions listed are degenerate in the sense that they are independent of at least one variable. The network diagram numbers for the degenerate functions have the suffix D. They are grouped together at the beginning of Table A3.

Table A3 shows all the optimum networks obtained for each function without imposing fan-in restrictions. However all the networks except that of function 026 are optimum even if the fan-in restriction that each gate can have at most three input interconnections is imposed. Function 026 needs one more gate if the above fan-in restriction is imposed. Optimum networks in this case are shown in Fig. A1.



Note that if a given function is symmetric in some variables, say  $a$  and  $b$ , then, among all the optimum networks obtainable by permuting these symmetric variables,  $a$  and  $b$ , only one network is listed in Table A2 and Table A3. The rest of networks can be obtained by simply exchanging the connection from the external variables according to the permutation of variables.

	0	1	2	3	4	5	6	7
0	-1*000	1*001	1*002	-1*003	3*002	-3*003	1*006	1*007
10	1*010	1*011	-1*012	1*013	-4*012	4*013	1*016	-1*017
20	2*002	-2*003	2*006	2*007	3*006	3*007	1*026	1*027
30	1*030	1*031	1*032	1*033	4*032	4*033	1*036	1*037
40	2*010	2*011	-6*012	6*013	2*030	2*031	6*032	6*033
50	1*050	1*051	1*052	1*053	1*054	1*055	1*056	1*057
60	-2*012	2*013	2*016	-2*017	2*032	2*033	2*036	2*037
70	6*054	6*055	6*056	6*057	-1*074	1*075	1*076	-1*077
100	3*010	3*011	3*030	3*031	-3*012	3*013	3*032	3*033
110	3*050	3*051	4*054	4*055	3*052	3*053	4*056	4*057
120	-5*012	5*013	5*032	5*033	3*016	-3*017	3*036	3*037
130	3*054	3*055	-3*074	3*075	3*056	3*057	3*076	-3*077
140	2*050	2*051	2*054	2*055	5*054	5*055	-2*074	2*075
150	1*150	1*151	1*152	1*153	3*152	3*153	1*156	1*157
160	2*052	2*053	2*056	2*057	5*056	5*057	2*076	-2*077
170	2*152	2*153	2*156	2*157	3*156	3*157	1*176	1*177
200	1*200	1*201	1*202	1*203	3*202	3*203	1*206	1*207
210	-1*210	1*211	1*212	1*213	4*212	4*213	1*216	1*217
220	2*202	2*203	2*206	2*207	3*206	3*207	1*226	1*227
230	1*230	-1*231	1*232	1*233	4*232	4*233	1*236	1*237
240	-2*210	2*211	6*212	6*213	2*230	-2*231	6*232	6*233
250	1*250	1*251	-1*252	1*253	1*254	1*255	1*256	-1*257
260	2*212	2*213	2*216	2*217	2*232	2*233	2*236	2*237
270	6*254	6*255	6*256	-6*257	1*274	1*275	1*276	1*277
300	-3*210	3*211	3*230	-3*231	3*212	3*213	3*232	3*233
310	3*250	3*251	4*254	4*255	-3*252	3*253	4*256	-4*257
320	5*212	5*213	5*232	5*233	3*216	3*217	3*236	3*237
330	3*254	3*255	3*274	3*275	3*256	-3*257	3*276	3*277
340	2*250	2*251	2*254	2*255	5*254	5*255	2*274	2*275
350	1*350	1*351	1*352	1*353	3*352	3*353	-1*356	1*357
360	-2*252	2*253	2*256	-2*257	5*256	-5*257	2*276	2*277
370	2*352	2*353	-2*356	2*357	-3*356	3*357	1*376	-1*377

#### Explanatory Example

Class of 321 is given by word at intersection of row 320 and column 1, 5\*213.

This says 321 is in class of 213 by permutation 5.

Negative permutation means the function is degenerate.

Permutation 1 is the identity

Permutation 2 is (abc)

Permutation 3 is (acb)

Permutation 4 is (bc)

Permutation 5 is (ac)

Permutation 6 is (ab)

Table A1. Equivalent classes of functions of three variables.

Table A2: Catalog of all the optimum NOR-AND networks  
of three variable functions.

Function (octal)	Functional expression	Network number	No. of gates		No. of inter- connections and connections	No. of levels
			NOR	AND		
000	0	1D	0	0	0	0
003	$a'b'$	5D	1	0	2	1
012	$a'c$	8D	2	0	3	2
		9D	1	1	3	2
017	$a'$	4D	1	0	1	1
074	$ab' + a'b$	13D	2	1	6	2
077	$a' + b'$	10D	1	1	3	2
210	$bc$	6D	0	1	2	1
231	$bc + b'c'$	14D	3	1	7	3
		15D	3	1	7	3
252	$c$	3D	0	0	0	0
257	$a' + c$	11D	3	0	4	3
		12D	2	1	4	3
356	$b + c$	7D	2	0	3	2
377	1	2D	0	0	0	0
001	$a'b'c'$	1	1	0	3	1
002	$a'b'c$	3	2	0	4	2
		7	1	1	4	2
006	$a'b'c + a'bc'$	15	2	1	7	2
007	$a'b' + a'c'$	8	1	1	4	2
010	$a'bc$	6	1	1	4	2
011	$a'bc + a'b'c'$	54	3	1	8	3
		61	3	1	8	3
013	$a'b' + a'c$	18	3	0	5	3

Function (octal)	Functional expression	Network number	No. of gates		No. of inter- connections and connections	No. of levels
			NOR	AND		
		22	2	1	5	3
016	$a'b + a'c$	4	2	0	4	2
026	$a'b'c + a'bc' + ab'c'$	68	2	3	13	2
027	$a'b' + b'c' + a'c'$	30	1	3	9	2
030	$ab'c' + a'bc$	70	4	1	10	3
		71	3	2	10	3
		88	4	1	10	4
031	$a'bc + b'c'$	85	4	1	9	4
032	$a'c + ab'c'$	28	2	2	9	2
033	$a'c + b'c'$	34	3	1	7	3
		44	2	2	7	3
036	$a'b + a'c + ab'c'$	29	2	2	10	2
037	$a' + b'c'$	17	1	2	6	2
050	$a'bc + ab'c$	27	3	1	8	2
		55	2	2	8	3
051	$a'b'c' + a'bc + ab'c$	79	3	2	11	3
052	$a'c + b'c$	11	2	1	5	2
		26	1	2	5	3
053	$a'b' + a'c + b'c$	36	3	1	8	3
054	$a'b + ab'c$	47	2	2	8	3
055	$a'b + a'c' + ab'c$	74	3	2	11	3
		78	3	2	11	3
056	$a'b + b'c$	12	2	1	6	2
057	$a' + b'c$	33	3	1	7	3
		43	2	2	7	3
075	$a'b + ab' + a'c'$	35	3	1	8	3
		45	2	2	8	3
076	$a'b + ab' + a'c$	14	2	1	7	2

Function (octal)	Functional expression	Network number	No. of gates		No. of inter- connections and connections	No. of levels
			NOR	AND		
150	$a'bc + ab'c + abc'$	80	3	2	11	3
151	$a'bc + ab'c + abc' + a'b'c'$	101	3	3	14	3
152	$a'c + b'c + abc'$	62	2	2	8	3
153	$a'c + a'b' + b'c + abc'$	76	3	2	11	3
156	$a'c + b'c + bc'$	13	2	1	7	2
157	$a' + b'c + bc'$	37	3	1	9	3
		46	2	2	9	3
176	$ab' + bc' + a'c$	16	2	1	8	2
177	$a' + b' + c'$	9	1	1	4	2
200	$abc$	2	0	1	3	1
201	$abc + a'b'c'$	52	3	1	9	3
202	$abc + a'b'c$	73	4	1	9	3
		77	4	1	9	3
		82	3	2	9	4
		90	3	2	9	4
203	$abc + a'b'$	50	3	1	8	3
206	$a'b'c + a'bc' + abc$	100	4	2	12	3
		104	4	2	12	4
		106	4	2	12	4
		109	4	2	12	4
		110	4	2	12	4
		111	4	2	12	4
		112	4	2	12	4
207	$a'b' + a'c' + abc$	81	3	2	9	3

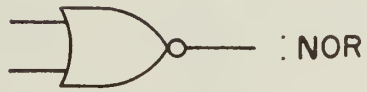
Function (octal)	Functional expression	Network number	No. of gates		No. of inter- connections and connections	No. of levels
			NOR	AND		
		89	3	2	9	4
		94	3	2	9	4
211	$bc + a'b'c'$	51	3	1	8	3
212	$a'c + bc$	31	4	0	6	3
		38	3	1	6	3
		64	3	1	6	4
		66	2	2	6	4
213	$a'b' + bc$	49	3	1	7	3
		56	3	1	7	3
216	$a'b + a'c + bc$	72	4	1	9	3
		86	4	1	9	4
		91	3	2	9	4
		95	4	1	9	4
		96	4	1	9	4
217	$a' + bc$	48	3	1	6	3
		67	2	2	6	4
226	$abc + ab'c' + a'bc' + a'b'c$	105	4	2	12	4
227	$a'b' + a'c' + b'c' + abc$	99	4	2	12	3
		102	4	2	12	4
230	$ab'c' + bc$	84	4	1	9	4
		87	3	2	9	4
		97	3	2	9	4
232	$a'c + bc + ab'c'$	92	4	1	9	4
		93	3	2	9	4
233	$a'b' + b'c' + bc$	57	3	1	8	3



Function (octal)	Functional expression	Network number	No. of gates		No. of inter- connections and connections	No. of levels
			NOR	AND		
236	$a'c + bc + a'b + ab'c'$	98	4	2	12	3
		103	4	2	12	4
		107	4	2	12	4
		108	4	2	12	4
237	$a' + bc + b'c'$	69	4	1	9	3
		83	3	2	9	4
250	$ac + bc$	10	3	0	5	2
		21	2	1	5	3
251	$ac + bc + a'b'c'$	59	3	1	8	3
253	$a'b' + c$	20	3	0	5	3
254	$ac + a'b$	32	4	0	7	3
		39	3	1	7	3
255	$ac + bc + a'c'$	58	3	1	8	3
256	$a'b + c$	63	4	0	6	4
		65	3	1	6	4
274	$ab' + ac + a'b$	40	3	1	8	3
275	$ab' + ac + a'b + a'c'$	60	3	1	9	3
276	$a'b + ab' + c$	41	3	1	9	3
277	$a' + b' + c$	23	2	1	5	3
350	$ab + ac + bc$	42	3	1	8	3
351	$ab + ac + bc + a'b'c'$	75	3	2	11	3
352	$ab + c$	25	2	1	5	3
353	$ab + a'b' + c$	53	3	1	8	3
357	$a' + b + c$	19	3	0	5	3
		24	2	1	5	3
376	$a + b + c$	5	2	0	4	2

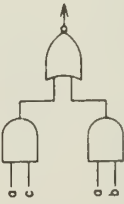









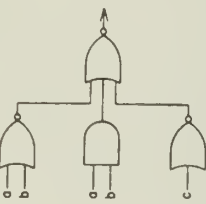
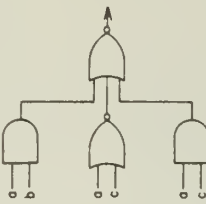
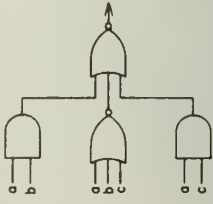
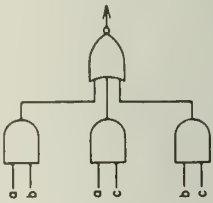
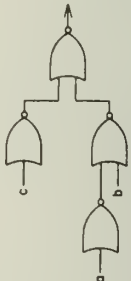
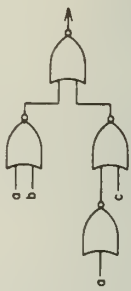


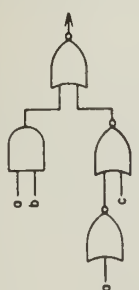
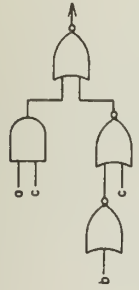
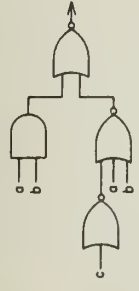
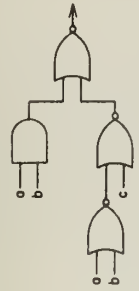
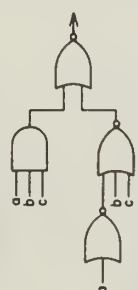
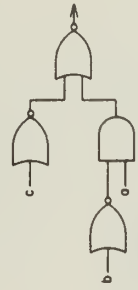
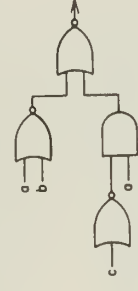
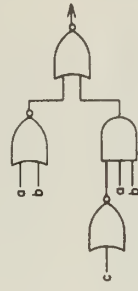
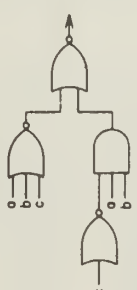
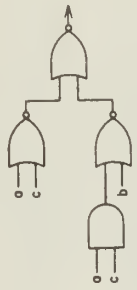
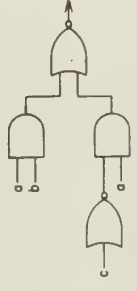
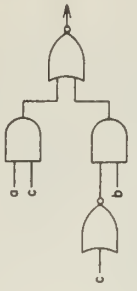
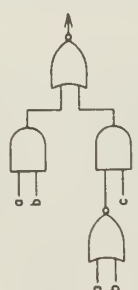
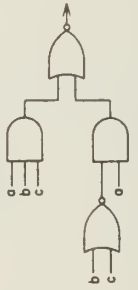
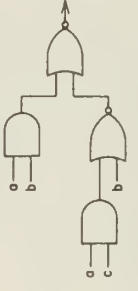
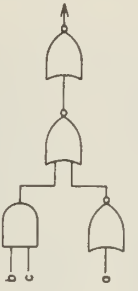
Table A3 List of all the optimum NOR-AND combination networks for three variable switching functions.






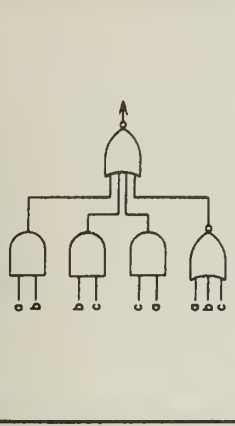
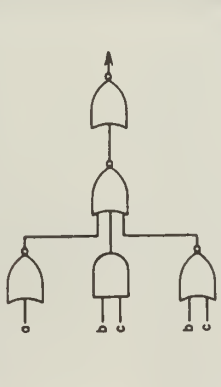
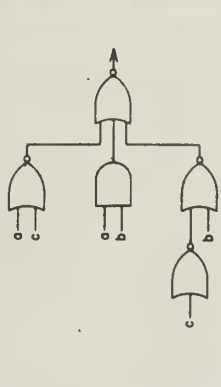
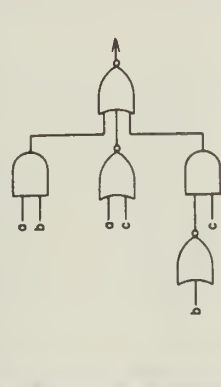
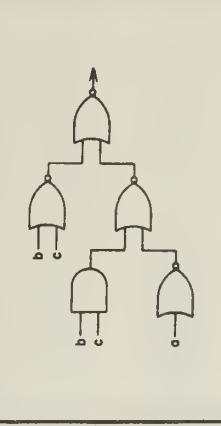
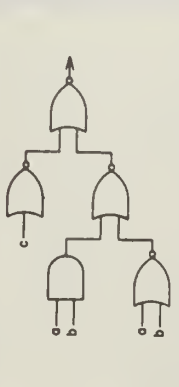
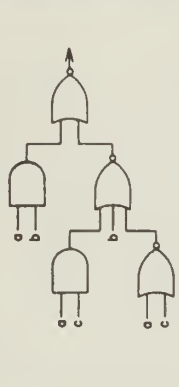
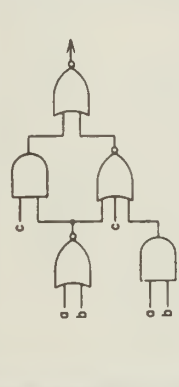
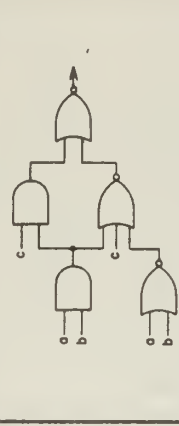
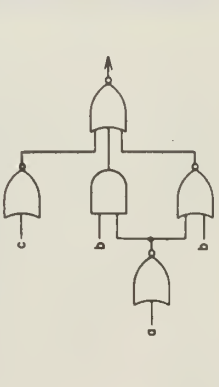
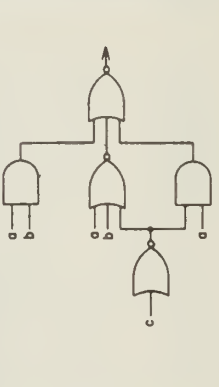
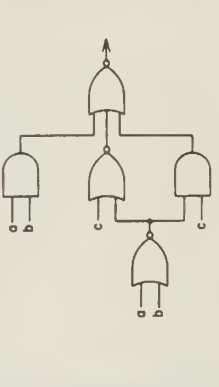
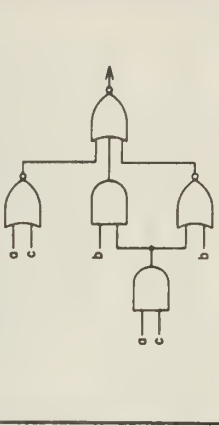
FUNCTION		FUNCTION		FUNCTION		FUNCTION		FUNCTION	
NO	ID	NO	2D	NO	377	NO	252	NO	017
	0		1		c		a		a'
5D	003	6D	210	7D	356	8D	012		
	a'b'		bc		b+c		a'c		
9D	012	10D	077	11D	257	12D	257		
	a'c		a'+b'		a'+c		a'+c		
13D	074	14D	231	15D	231				
	a'b+a'b		bc+b'c'		bc+b'c'				

NO	FUNCTION	NO	FUNCTION	NO	FUNCTION	NO	FUNCTION				
1	001	$a'b'c'$	2	200	$abc$	3	002	$a'b'c$	4	016	$a'b+a'c$
5	376	$a+b+c$	6	010	$a'bc$	7	002	$a'b'c$	8	007	$a'b+a'c$
9	177	$a'b'+c'$	10	250	$ac+bc$	11	052	$a'c+b'c$	12	056	$a'b+b'c$
13	156	$a'c+bc+bc'$	14	076	$a'b+ab'+a'c$	15	006	$a'bc+a'bc'$	16	176	$ab'+bc'+a'c$

NO	FUNCTION	NO	FUNCTION	NO	FUNCTION	NO	FUNCTION				
17	037	$a + b'c$	18	013	$a'b + a'c$	19	357	$a + b + c$	20	253	$a'b + c$
											
21	250	$ac + bc$	22	013	$a'b + a'c$	23	277	$a + b + c$	24	357	$a + b + c$
											
25	352	$ab + c$	26	052	$d'c + b'c$	27	050	$a'bc + abc$	28	032	$a'c + abc'$
											
29	036	$a'b + a'c + ab'c'$	30	027	$a'b + b'c' + a'c'$	31	212	$a'c + bc$	32	254	$ac + a'b$
											

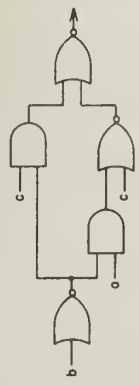
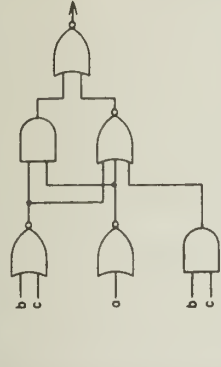
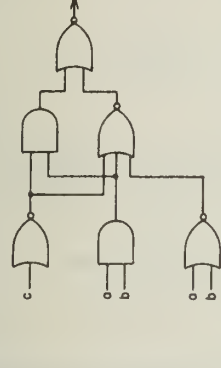
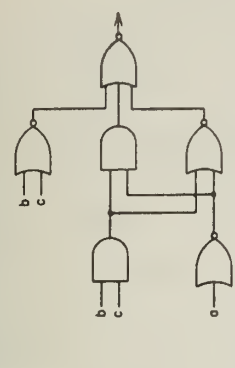
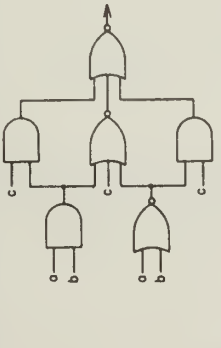
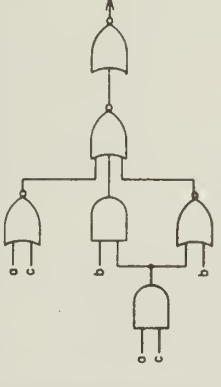
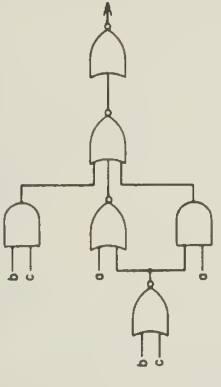
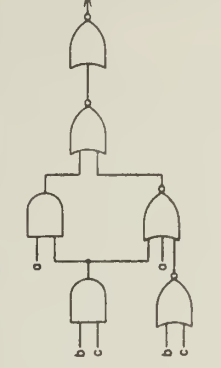
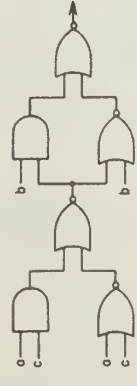

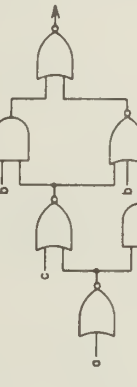

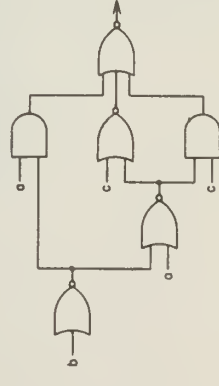
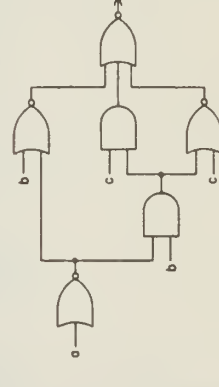
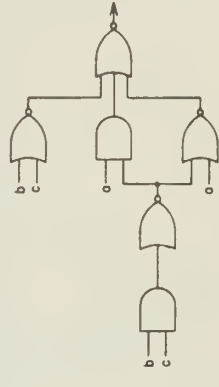
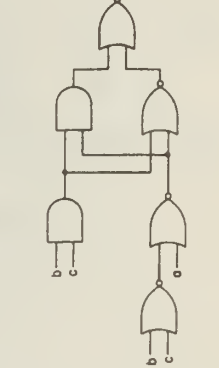
NO	FUNCTION	NO	FUNCTION	NO	FUNCTION	NO	FUNCTION
33	057	$a' + b'c$		34	033	$dc + b'c'$	
35	075	$a'b + ab' + d'c'$		36	053	$a'b' + a'c + b'c$	
37	157	$a' + bc + bc'$		38	212	$d'c + bc$	
39	254	$ac + a'b$		40	274	$ab' + ac + a'b$	
41	276	$a'b + ab' + c$		42	350	$ab + ac + bc$	
43	057	$d' + bc$		44	033	$a'c + b'c'$	
45	075	$a'b + ab' + d'c'$		46	157	$d + b'c + bc'$	
47	054	$d'b + abc$		48	217	$a' + bc$	

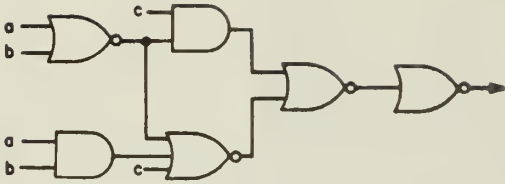
FUNCTION		FUNCTION		FUNCTION		FUNCTION	
NO		NO		NO		NO	
49	213	50	203	51	211	52	201
	$a'b+bc$		$abc+a'b$		$bc+abc'$		$abc+a'bc'$
53	353	54	011	55	050	56	213
	$ab+ab'+c$		$a'bc+a'b'c'$		$a'bc+abc$		$a'b+bc$
57	233	58	255	59	251	60	275
	$a'b'+bc'+bc$		$ac+bc+dc'$		$ac+bc+a'b'c'$		$ab+ac+ab'+d'$
61	011	62	152	63	256	64	212
	$a'bc+a'b'c'$		$dc+b'c+abc'$		$db+c$		$a'c+bc$

FUNCTION		FUNCTION		FUNCTION		FUNCTION	
NO		NO		NO		NO	
65	256	66	212	67	217	68	026
	$a'b + c$		$a'c + bc$		$a' + bc$		$a'bc + a'bc' + abc'$
							
69	237	70	030	71	030	72	216
	$a' + bc + bc'$		$ab'c' + abc$		$ab'c' + abc$		$a'b + bc' + bc$
							
73	202	74	055	75	351	76	153
	$abc + a'bc$		$a'b + a'c' + abc$		$ab + ac + bc + a'bc'$		$ac + a'b' + bc' + abc'$
							
77	202	78	055	79	051	80	150
	$abc + a'bc$		$a'b + a'c' + abc$		$a'bc' + abc + abc$		$abc + abc' + abc'$
							

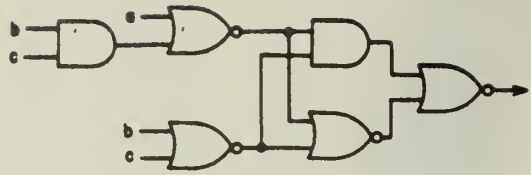


FUNCTION		FUNCTION		FUNCTION		FUNCTION		FUNCTION	
NO		NO		NO		NO		NO	
81	$a'b + a'c + abc$	82	$abc + a'bc$	83	$a + bc + bc'$	84	$ab'c' + bc$	85	$abc + bc'$
86	$a'b + a'c + bc$	87	$ab'c' + bc$	88	$ab'c' + a'bc$	89	$a'b + a'c' + abc$	90	$abc + a'bc$
91	$a'b + a'c + bc$	92	$a'c + bc + ab'c'$	93	$a'c + bc + ab'c'$	94	$a'b + a'c' + abc$	95	$a'b + a'c + bc$
96	$a'b + a'c + bc$	97	$a'b + a'c + bc$	98	$a'b + a'c + bc$	99	$a'b + a'c + bc$	100	$a'b + a'c + bc$

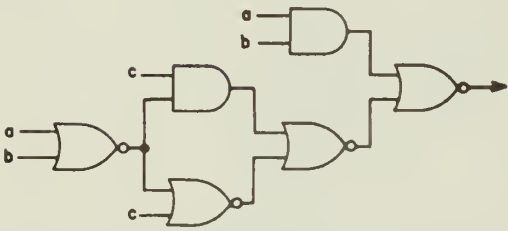
NO	FUNCTION	NO	FUNCTION	NO	FUNCTION		
97	230 $a\bar{b}\bar{c}+bc$ 	98	236 $a\bar{c}+bc+a\bar{b}+abc$ 	99	227 $a\bar{b}+a\bar{c}+b\bar{c}+abc$ 	100	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 
101	151 $a\bar{b}c+abc+abc'+ab\bar{c}'$ 	102	227 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 	103	236 $a\bar{c}+bc+a\bar{b}+abc'$ 	104	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 
105	226 $abc+ab\bar{c}'+ab\bar{c}+ab\bar{c}$ 	106	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 	107	236 $a\bar{c}+bc+a\bar{b}+abc'$ 	108	236 $a\bar{c}+bc+a\bar{b}c+ab\bar{c}$ 
109	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 	110	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 	111	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 	112	206 $a\bar{b}\bar{c}+a\bar{b}c+abc$ 



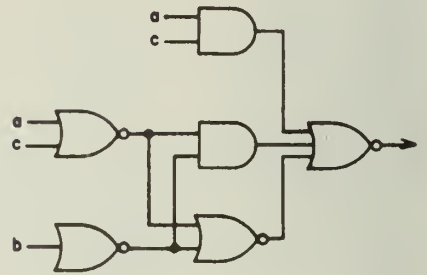
(1)



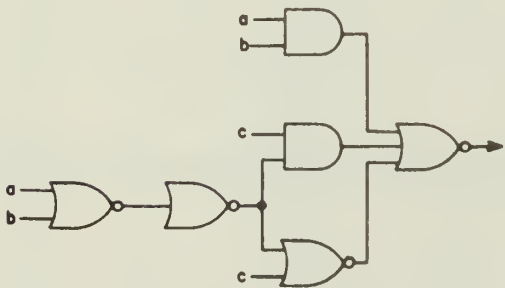
(2)



(3)



(4)

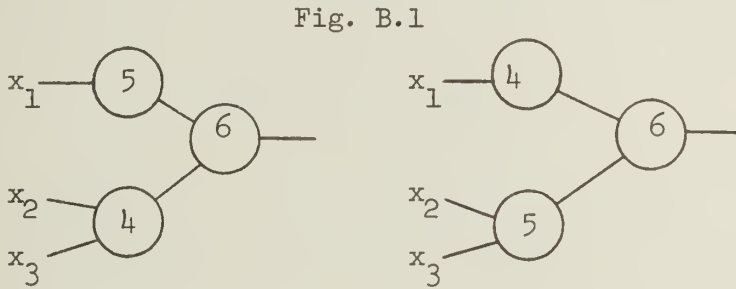


(5)

Fig. A1 Optimum networks of function 026 when the fan-in is limited to three.

APPENDIX B: Ordering of Interconnections

In the feed-forward network formulation gates are uniquely labeled 1, 2, ..., R. Usually there are many representations for the same configuration of the optimal network simply by permuting the labels on gates. For example the two networks in Fig. B.1 are equivalent under permutation of gate labels.



In order to reduce this obvious redundancy many inequalities are formulated to order the gates.

For an example of such inequalities a list of inequalities for the 6 gate feed-forward NOR network will be presented.

Since the inequalities were not generated systematically, the list is not complete.

Let us denote  $(1 - \varphi_j)$  by  $\bar{\varphi}_{ij}$  where  $\varphi_{ij}$  is 1 if there is an interconnection from gate  $i$  to gate  $j$  and 0 otherwise.

Note that the inequalities are for 6 gate NOR networks with unlimited fan-ins and fan-outs. The inequalities corresponding to an R gate NOR network with restrictions on fan-ins and fan-outs can be written in an analogous way.

1. Order Inputs to the R-th Gate (i.e. output gate)

The ordering of connecting gates 1 through R - 1 to the output gate R can be specified as  $\varphi_{iR} \geq \varphi_{jR}$  for all  $i > j$ . This follows from that fact that any circuit with  $\varphi_{kR} \geq \varphi_{lR}$  and  $k < l$  can be obtained from a circuit in which  $\varphi_{iR} \geq \varphi_{jR}$  for all  $i > j$  by permutation of gate labels,  $k$  and  $l$ .

$$\varphi_{46} \leq \varphi_{56}$$

$$\varphi_{36} \leq \varphi_{46}$$

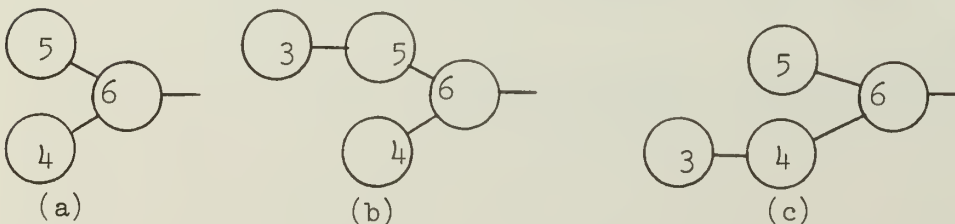
$$\varphi_{26} \leq \varphi_{36}$$

$$\varphi_{16} \leq \varphi_{26}$$

2. Ordering the Outputs of All Isolated Gates

As was defined previously an isolated gate is a gate which has none of variables,  $v_i$ 's and  $\varphi_{jk}$ 's representing its inputs and outputs specified to 1. When the isolated gate is connected to the subnetwork, the isolated gate can often be connected in several ways. Furthermore the networks resulting from these different ways of connecting the gate are often equivalent with respect to permutation of the gate labels. In Fig. B.2 networks (b) and (c) give an example of two different ways of connecting the isolated gate 3.

Fig. B.2



By examining Fig. B.2 we can see that gates 4 and 5 of the subnetwork consisting of gates 4,5, and 6 are equivalent with respect to permutation of gate labels and the connection of gate 3 to gate 4 or 5 (i.e. network (b) is equivalent to network (c) if gate labels 4 and 5 are permuted). Thus we can restrict  $\varphi_{34} \leq \varphi_{35}$  for the given subnetwork consisting of gates 4,5 and 6 to prohibit network (c).

The subnetwork (a) in Fig. B.2 can be specified as

$$\varphi_{36} + \bar{\varphi}_{46} + \bar{\varphi}_{56} \tag{1}$$

for the following reason. The sum in expression (1) assumes the value zero if and only if the gate 4,5 and 6 are connected as shown in Fig. B.2 (a) and otherwise it is 1 or greater. Thus we can order the pair of interconnections  $\varphi_{35}$  and  $\varphi_{34}$  under the assumption that the subnetwork of Fig. 3.2 (a) exists. This conditional ordering of gate output interconnections can be expressed by the following inequality

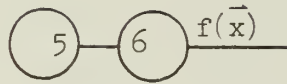
$$\varphi_{34} \leq \varphi_{35} + \bar{\varphi}_{36} + \bar{\varphi}_{46} + \bar{\varphi}_{56}. \tag{2}$$

Because if the subnetwork exists, the last three terms of (2) becomes 0 and (2) becomes  $\varphi_{34} \leq \varphi_{35}$ . Let us call  $\varphi_{34} \leq \varphi_{35}$  of expression (2) as the ordering portion and  $+\varphi_{36} + \bar{\varphi}_{46} + \bar{\varphi}_{56}$  as the subnetwork specification portion. Note that the subnetwork specification portion will be positive and expression (2) will be non-restrictive if the subnetwork described by  $\varphi_{36} + \bar{\varphi}_{46} + \bar{\varphi}_{56}$  does not exists.

All the following inequalities in this section can be similarly interpreted by first examining the subnetwork specification portion

of the inequality and then observing the ordering required on the pair of  $\varphi_{ij}$  variables.

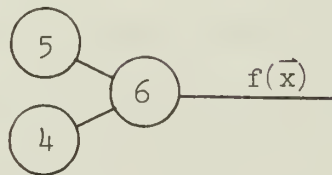
a)



This subnetwork is denoted by  $\bar{\varphi}_{56} + \varphi_{46}$ . Assuming this subnetwork we can get the following inequalities in terms of isolated gates, according to the above discussion.

<u>Ordering</u>	<u>Subnetwork</u>
$\varphi_{35} \leq \varphi_{45}$	+ $\bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{25} \leq \varphi_{35}$	+ $\bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{15} \leq \varphi_{25}$	+ $\bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{23} \leq \varphi_{24}$	+ $\bar{\varphi}_{35} + \bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{13} \leq \varphi_{14}$	+ $\bar{\varphi}_{25} + \bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{12} \leq \varphi_{13}$	+ $\bar{\varphi}_{25} + \bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{14} \leq \varphi_{24}$	+ $\bar{\varphi}_{35} + \varphi_{12} + \bar{\varphi}_{56} + \varphi_{46}$
$\varphi_{13} \leq \varphi_{14}$	+ $\bar{\varphi}_{35} + \bar{\varphi}_{23} + \bar{\varphi}_{56} + \varphi_{46}$

b)



This subnetwork is denoted by  $\bar{\varphi}_{46} + \varphi_{36}$ .



Ordering

$$\varphi_{34} \leq \varphi_{35}$$

$$\varphi_{24} \leq \varphi_{25}$$

$$\varphi_{14} \leq \varphi_{15}$$

$$\varphi_{15} \leq \varphi_{25}$$

$$\varphi_{14} \leq \varphi_{15}$$

$$\varphi_{12} \leq \varphi_{13}$$

$$\varphi_{12} \leq \varphi_{13}$$

Subnetwork

$$+ \bar{\varphi}_{46} + \varphi_{36}$$

$$+ \varphi_{34} + \bar{\varphi}_{46} + \varphi_{36}$$

$$+ \bar{\varphi}_{24} + \bar{\varphi}_{34} + \bar{\varphi}_{46} + \varphi_{36}$$

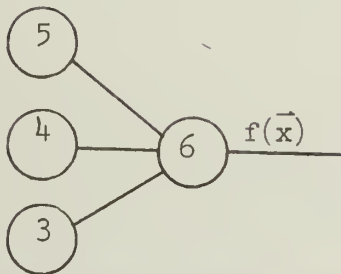
$$+ \varphi_{12} + \bar{\varphi}_{46} + \varphi_{36}$$

$$+ \varphi_{23} + \bar{\varphi}_{35} + \bar{\varphi}_{24} + \varphi_{25} + \bar{\varphi}_{46} + \varphi_{36}$$

$$+ \varphi_{15} + \varphi_{23} + \bar{\varphi}_{35} + \bar{\varphi}_{24} + \varphi_{25} + \bar{\varphi}_{46} + \varphi_{36}$$

$$+ \bar{\varphi}_{14} + \varphi_{23} + \bar{\varphi}_{35} + \bar{\varphi}_{24} + \varphi_{25} + \bar{\varphi}_{46} + \varphi_{36}$$

c)



This subnetwork is denoted by  $\bar{\varphi}_{36} + \varphi_{26}$ .

Ordering

$$\varphi_{24} \leq \varphi_{25}$$

$$\varphi_{23} \leq \varphi_{24}$$

$$\varphi_{13} \leq \varphi_{14}$$

$$\varphi_{14} \leq \varphi_{15}$$

$$\varphi_{13} \leq \varphi_{14}$$

$$\varphi_{14} \leq \varphi_{15}$$

$$\varphi_{15} \leq \varphi_{25}$$

Subnetwork

$$+ \bar{\varphi}_{36} + \varphi_{26}$$

$$+ \bar{\varphi}_{36} + \varphi_{26}$$

$$+ \varphi_{24} + \bar{\varphi}_{36} + \varphi_{26}$$

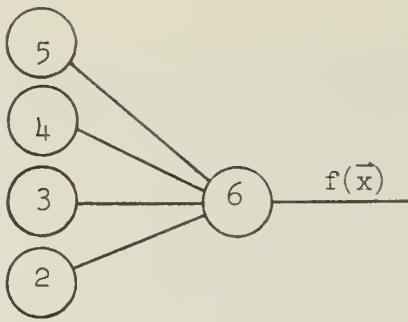
$$+ \bar{\varphi}_{24} + \bar{\varphi}_{36} + \varphi_{26}$$

$$+ \bar{\varphi}_{23} + \bar{\varphi}_{36} + \varphi_{26}$$

$$+ \bar{\varphi}_{23} + \bar{\varphi}_{36} + \varphi_{26}$$

$$+ \varphi_{12} + \bar{\varphi}_{36} + \varphi_{26}$$

d)



This subnetwork is denoted by  $\bar{\varphi}_{26} + \varphi_{16}$ .

Ordering

$$\varphi_{14} \leq \varphi_{15}$$

$$\varphi_{13} \leq \varphi_{14}$$

$$\varphi_{12} \leq \varphi_{13}$$

Subnetwork

$$+ \bar{\varphi}_{26} + \varphi_{16}$$

$$+ \bar{\varphi}_{26} + \varphi_{16}$$

$$+ \bar{\varphi}_{26} + \varphi_{16}$$

### 3. Ordering the Outputs of a Single Gate

If two gates  $i$  and  $i+1$  are not connected (i.e.  $\varphi_{i, i+1} = 0$ ), the same network configuration can be obtained by permuting gate labels  $i$  and  $i+1$ . To prohibit this permutation restrictions are placed on the outputs of gates  $i$  and  $i+1$ . Let us examine the following inequality using the terms, ordering portion and subnetwork specification portion of an inequality which were defined in the previous section:

$$4\varphi_{26} + 2\varphi_{25} + \varphi_{24} \leq 4\varphi_{36} + 2\varphi_{35} + \varphi_{34} + 8\varphi_{23} \quad (3)$$

The subnetwork specification portion of expression (3) is  $8\varphi_{23}$ . The ordering portion is  $4\varphi_{26} + 2\varphi_{25} + \varphi_{24} \leq 4\varphi_{36} + 2\varphi_{35} + \varphi_{34}$  which expresses the restrictions

$$\varphi_{26} \leq \varphi_{36},$$

$$\varphi_{25} \leq \varphi_{35}, \quad \text{if } \varphi_{26} = \varphi_{36} \text{ and}$$

$$\varphi_{24} \leq \varphi_{34} \quad \text{if } \varphi_{25} = \varphi_{35} \text{ and } \varphi_{26} = \varphi_{36},$$

given the subnetwork described by  $\varphi_{23}$ . Note that  $\varphi_{23}$  is 1 if the subnetwork does not exist and hence inequality (3) will be non-restrictive and the outputs of gates 2 and 3 will not be ordered.

The remaining inequalities of this section can be similarly interpreted.

<u>Ordering</u>	<u>Subnetwork</u>
$2\varphi_{36} + \varphi_{35} \leq 2\varphi_{46} + \varphi_{45}$	+ $4\varphi_{34}$
$4\varphi_{15} + 2\varphi_{14} + \varphi_{13} \leq 4\varphi_{25} + 2\varphi_{24} + \varphi_{23}$	+ $8\varphi_{12}$
$4\varphi_{26} + 2\varphi_{25} + \varphi_{24} \leq 4\varphi_{36} + 2\varphi_{35} + \varphi_{34}$	+ $8\varphi_{23}$





MAR 3 5 1973













UNIVERSITY OF ILLINOIS-URBANA



3 0112 045402051